UNIVERSIDADE FEDERAL DE OURO PRETO

INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS

DEPARTAMENTO DE COMPUTAÇÃO

Daniel Pinto Moreira

PRÁTICAS DE SRE APLICADAS À INFRAESTRUTURA DE REDE: IMPACTOS DA APLICAÇÃO DE PRÁTICAS DE SRE NA IDENTIFICAÇÃO E RESPOSTA A INCIDENTES EM INFRAESTRUTURA DE TI.

Ouro Preto, MG
2025

Daniel Pinto Moreira

PRÁTICAS DE SRE APLICADAS À INFRAESTRUTURA DE REDE: IMPACTOS DA APLICAÇÃO DE PRÁTICAS DE SRE NA IDENTIFICAÇÃO E RESPOSTA A INCIDENTES EM INFRAESTRUTURA DE TI.

Monografia II apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Carlos Frederico Marcelo da Cunha Cavalcanti

Ouro Preto, MG
2025

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

DANIEL PINTO MOREIRA

Práticas de SRE aplicadas à infraestrutura de rede: Impactos da aplicação de práticas de SRE na identificação e resposta a incidentes em infraestrutura de TI

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 08 de Setembro de 2025.

Membros da banca

Doutor Carlos Frederico Marcelo da Cunha Cavalcanti (Orientador) - Universidade Federal de Ouro Preto
Doutor Fernando Cortez Sica (Examinador) - Universidade Federal de Ouro Preto
Doutor Ricardo Augusto Rabelo Oliveira (Examinador) - Universidade Federal de Ouro Preto

Prof. Dr. Carlos Frederico Marcelo da Cunha Cavalcanti, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 28/10/2025.

Documento assinado eletronicamente por **Carlos Frederico Marcelo da Cunha Cavalcanti**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 18/12/2025, às 16:08, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **1034992** e o código CRC **DC7B0599**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591692 - www.ufop.br

Agradecimentos

Em primeiro lugar, agradeço a Deus por me guiar em cada passo desta caminhada e por me conceder força e perseverança. Aos meus pais, Carlos e Zeli, registro meu imenso agradecimento pelo apoio incondicional, carinho, paciência, dedicação, fé e amor. Vocês são minha fonte de inspiração! Estendo também minha gratidão à minha irmã Amanda, cujo companheirismo e carinho tornaram essa jornada ainda mais significativa. Amo vocês!

À minha esposa, Lauana, não há palavras suficientes para expressar o quanto sou grato por ter alguém tão especial ao meu lado. Seu carinho e apoio incondicional, aliados às palavras de perseverança e coragem, fizeram de mim uma pessoa melhor. Você é minha paz, minha luz! Te amo!

À Creusa, minha segunda mãe, deixo o meu muito obrigado por todos os ensinamentos, carinho e dedicação. Amo você!

À minha família, meu profundo agradecimento por terem sido exemplo de união, companheirismo e carinho ao longo de toda a minha trajetória.

Aos meus amigos, agradeço de coração por todos os momentos compartilhados. Sem a parceria de vocês, essa caminhada teria sido muito mais difícil.

Expresso ainda minha gratidão ao professor Carlos Frederico por sua orientação, ensinamentos, apoio e incentivo ao longo da graduação e durante a elaboração desta monografia. Aos professores do DECOM, meu muito obrigado.

A todos que cruzaram o meu caminho e, de alguma forma, contribuíram para este momento, deixo aqui minha mais sincera gratidão. Serei eternamente agradecido a cada um de vocês por esta conquista. Que possamos continuar trilhando caminhos de sucesso e felicidade juntos. Obrigado por fazerem parte da minha vida!

“O Senhor é o meu pastor e nada me faltará.”

— Salmo 23:1

Resumo

A crescente complexidade das infraestruturas de redes corporativas, somada à dependência de serviços críticos e distribuídos, exige novas abordagens para garantir disponibilidade, confiabilidade e eficiência operacional. Nesse contexto, o presente trabalho investiga a aplicação de práticas de SRE *(Site Reliability Engineering)* em redes de médio porte, com foco na detecção e resposta a incidentes. O estudo adota como base os pilares da observabilidade — métricas, *logs* e *traces* — e implementa ferramentas consolidadas no mercado, como Zabbix, Graylog e Grafana, para compor um ecossistema de monitoramento proativo. A metodologia incluiu o mapeamento do ambiente, a definição de indicadores e objetivos de nível de serviço (SLIs e SLOs), a instrumentação da infraestrutura e a coleta de dados, permitindo análises comparativas e a identificação de oportunidades de automação e eliminação de tarefas repetitivas (TOIL). Os resultados obtidos demonstraram que a integração de práticas de SRE contribuiu para a redução do tempo de indisponibilidade, aumento da capacidade de resposta das equipes de TI e maior resiliência da rede diante de incidentes inesperados. Além disso, o trabalho reforça a importância do uso de relatórios de disponibilidade e análises pós-incidente (postmortem) como mecanismos de aprendizado contínuo e aprimoramento dos processos. Conclui-se que a adoção de SRE em ambientes de médio porte é viável e traz impactos significativos na maturidade da gestão de infraestrutura, servindo como referência para futuras implementações em contextos similares.

Palavras-chave: SRE, Observabilidade, Redes, Resiliência, Incidentes, Automação.

Abstract

The increasing complexity of corporate network infrastructures, combined with the reliance on critical and distributed services, demands new approaches to ensure availability, reliability, and operational efficiency. In this context, this work investigates the application of Site Reliability Engineering (SRE) practices in medium-sized networks, focusing on incident detection and response. The study is grounded on the pillars of observability — metrics, logs, and traces — and implements well-established tools such as Zabbix, Graylog, and Grafana to build a proactive monitoring ecosystem. The methodology involved environment mapping, definition of Service Level Indicators (SLIs) and Service Level Objectives (SLOs), infrastructure instrumentation, and data collection, enabling comparative analyses and the identification of opportunities for automation and toil reduction. The results demonstrated that integrating SRE practices contributed to reducing downtime, enhancing the response capacity of IT teams, and strengthening network resilience against unexpected incidents. Furthermore, the work highlights the relevance of availability reports and postmortem analyses as mechanisms for continuous learning and process improvement. It is concluded that adopting SRE in medium-sized environments is feasible and significantly impacts infrastructure management maturity, serving as a reference for future implementations in similar contexts.

Keywords: SRE, Observability, Networks, Resilience, Incidents, Automation.

Lista de abreviaturas e siglas

TI	Tecnologia da Informação
SaaS	Software as a Service
SRE	Site Reliability Engineering
SLA	Service Level Agreement
SLO	Service Level Objective
SLI	Service Level Indicator
IEEE	Institute of Electrical and Electronics Engineers
CPU	Central Processing Unit
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
SNMP	Simple Network Management Protocol
MIB	Management Information Base
OID	Object Identifier
IETF	Internet Engineering Task Force
TOIL	Tasks Often Involving Legwork
RFC	Request for Comments
IPMI	Intelligent Plataform Management Interface
IoT	Internet of Things
API	Application Programming Interface
GELF	Graylog Extended Log Format
JSON	JavaScript Object Notation
VPN	Virtual Private Network
Cobit	Control Objectives for Information and related Technology
PCN	Plano de continuidade de negócios

ERP	Enterprise Resource Planning
DoS	Denial of Service
SMA	Simple Movie Average
TAR	Trend-Aware Regression
AP	Access Point
VM	Virtual Machine
ICMP	Internet Control Message Protocol

Lista de símbolos

Figura 2.3 Evolução do protocolo SNMP

Figura 2.3 Estrutura MIB

Figura 4.1 Topologia da rede

Figura 4.12 Graylog

Figura 4.3 Dashboard - Links de internet

Figura 4.4 Dashboard - Link - Filiais

Figura 4.5 Dashboard - Link - SaaS

Figura 4.6 Dashboard - Serviços - SaaS

Figura 4.7 Dashboard - Switches

Figura 4.8 Dashboard - Containers

Figura 4.9 Dashboard - Problemas ativos

Figura 4.10 Dashboard - Falhas de login

Figura 4.13 Relatório Postmortem

Figura 5.1 Total de chamados abertos por mês

Figura 5.2 Total de chamados de impressora por mês

Figura 5.3 Dashboard - Monitoramento do uso de CPU

Figura 5.4 Dashboard - Monitoramento da partição de imagens

Figura 5.5 Dashboard - Falhas de login

Figura 4.11 Relatório de Disponibilidade

Sumário

1 Introdução 1
1.1 Justificativa 1
1.2 Objetivos 2
1.2.1 Principal 2
1.2.2 Específicos 2
1.3 Organização do Trabalho 2
2 Revisão Bibliográfica 4
2.1 Fundamentação Teórica 4
2.1.1 SRE 4
2.1.2 Observabilidade 5
2.1.3 Monitoramento 6
2.1.4 Monitoramento x Observabilidade 7
2.1.5 Níveis de Serviços 7
2.1.6 Error Budget 9
2.1.7 Eliminação de TOIL 9
2.1.8 Postmortem 10
2.1.9 Maturidade da rede 10
2.1.10 SNMP 10
2.1.11 Management Information Base (MIB) e Object Identifier (OID) 12
2.1.12 Zabbix 12
2.1.13 Graylog 13
2.1.14 Grafana 13
3 Trabalhos Relacionados 15
4 Desenvolvimento 18
4.1 Metodologia Científica 18
4.1.1 Mapeamento do ambiente 18
4.1.2 Definição de métricas, SLOs e SLIs 20
4.1.3 Instrumentação da infraestrutura 22
4.1.3.1 Configuração das Ferramentas de Observabilidade 22
4.1.3.2 Implantação e Configuração do Zabbix 22
4.1.3.3 Implantação e Configuração do Graylog 23
4.1.3.4 Implantação e Configuração do Grafana 23
4.1.4 Coleta de Dados 24
4.1.5 Eliminação de TOIL 24
4.1.6 Dashboards dos dados obtidos 25
5 Resultados 29

6 Conclusão 32

Referências 33
APÊNDICE A Metodologia 35

1 Introdução

No mundo atual, a interconectividade proporcionada pelas redes de computadores desempenha um papel vital no funcionamento das organizações. Manter sistemas e serviços disponíveis para colaboradores e clientes tornou-se uma tarefa cada vez mais complexa. A crescente dependência de serviços de terceiros, como *links* de internet e, mais recentemente, soluções do tipo *Software as a Service* (SaaS), aliada à diversidade de serviços e à necessidade de garantir a integridade e o sigilo dos dados, aumenta substancialmente a complexidade e a responsabilidade das equipes envolvidas.

A demora na identificação e na resposta a incidentes na infraestrutura de rede pode impactar significativamente o negócio, ocasionando interrupções nos serviços, perda de dados e até prejuízos financeiros consideráveis. Para mitigar esses riscos, as equipes de infraestrutura adotam diferentes abordagens e ferramentas voltadas ao monitoramento e à identificação de problemas. No entanto, muitas dessas soluções têm caráter reativo e não conseguem lidar adequadamente com situações imprevisíveis ou falhas emergentes em ambientes cada vez mais distribuídos.

Originalmente criado para operações de sistemas em grande escala, o *Site Reliability Engineering*(SRE) traz práticas que ajudam a tornar ambientes mais confiáveis, previsíveis e gerenciáveis. O SRE, quando adaptado à infraestrutura de TI, apresenta uma abordagem promissora para enfrentar esses desafios. Ao aplicar práticas de SRE com foco na infraestrutura, os profissionais de TI passam a ter uma visão mais detalhada do ambiente, o que facilita a identificação de padrões, tendências e falhas iminentes. Além disso, essa abordagem contribui para detectar o uso excessivo de recursos, bem como identificar pontos passíveis de melhoria. Dessa forma, torna-se possível agir de maneira proativa, antecipando problemas antes que causem interrupções nos serviços.

Diante disso, este trabalho tem como objetivo explorar a aplicação de práticas de SRE na detecção e resposta a incidentes de infraestrutura em redes corporativas de médio porte. A pesquisa buscará desenvolver e validar um modelo de integração entre ferramentas de observabilidade e processos de gestão de incidentes, de modo a contribuir para a melhoria da eficiência operacional, redução do tempo de inatividade e fortalecimento da resiliência da infraestrutura de TI.

1.1 Justificativa

A escolha deste tema se justifica pela necessidade de aprimorar os processos de identificação e resposta a incidentes em redes corporativas de médio e pequeno porte. Embora menores do que as infraestruturas de grandes empresas, essas redes enfrentam desafios significativos, especialmente em razão da escassez de investimentos e da exigência de manter a continuidade

das operações.

A integração de práticas de SRE à gestão de infraestrutura apresenta-se como uma solução promissora. Ao proporcionar uma visibilidade contínua e detalhada sobre o estado dos sistemas, a adoção desta abordagem permite que as equipes de TI identifiquem rapidamente anomalias e respondam de forma mais eficaz aos incidentes. Isso contribui diretamente para o aumento da resiliência da infraestrutura, além de reduzir o tempo de inatividade e os custos associados a falhas inesperadas. Dessa forma, ao focar neste tema, espera-se contribuir para o desenvolvimento de metodologias mais eficazes de detecção e resposta a incidentes.

1.2 Objetivos

1.2.1 Principal

O objetivo geral deste trabalho é explorar e implementar práticas de SRE em uma infraestrutura de TI, com ênfase na resiliência da infraestrutura de rede. Busca-se, com isso, aprimorar a gestão de incidentes, abrangendo a detecção, a resposta, o aprendizado e a geração de relatórios de disponibilidade e desempenho.

1.2.2 Específicos

1. Desenvolver um modelo de integração entre práticas de SRE e a gestão de incidentes de infraestrutura, adaptado às características da rede em estudo, incluindo a identificação de ferramentas, métricas e métodos para a coleta e análise de dados.
2. Implementar e configurar as ferramentas selecionadas para a coleta e análise de dados, com a criação de *dashboards* voltados ao acompanhamento do desempenho em tempo real.
3. Validar a eficiência do modelo e das ferramentas propostas na identificação e resolução de incidentes.
4. Desenvolver relatórios de disponibilidade, recomendações e recalibrar métricas e alertas caso necessário.
5. Avaliar a viabilidade de automatizar a solução de incidentes recorrentes ou previamente conhecidos.
6. Analisar formas de mitigar erros encontrados.

1.3 Organização do Trabalho

Este trabalho está estruturado da seguinte forma:

- **Capítulo 1 – Introdução:** apresenta o contexto do trabalho, destacando os desafios atuais enfrentados pelas equipes de infraestrutura de TI, especialmente em redes corporativas. São discutidos os objetivos da pesquisa, sua justificativa e a organização geral da monografia.
- **Capítulo 2 – Revisão Bibliográfica:** apresenta os conceitos fundamentais relacionados ao tema, como observabilidade, monitoramento, níveis de serviço, maturidade de rede, protocolos SNMP e descreve as principais ferramentas utilizadas no trabalho, como Zabbix, Graylog e Grafana.
- **Capítulo 3 – Metodologia:** detalha a abordagem científica adotada e os procedimentos metodológicos para coleta e análise de dados.
- **Capítulo 4 – Trabalhos Relacionados:** realiza uma análise de estudos e projetos anteriores que abordam práticas de monitoramento, observabilidade e resiliência de redes. Este capítulo busca identificar as principais abordagens adotadas na literatura, destacando similaridades e diferenças em relação à proposta deste trabalho, além de evidenciar lacunas e oportunidades para avanços no tema.
- **Capítulo 5 – Resultados e Discussões:** discute os resultados obtidos até o momento, com base na aplicação prática das ferramentas e princípios de SRE, e propõe análises preliminares sobre os impactos observados.
- **Capítulo 6 – Conclusão e Trabalhos Futuros:** apresenta as considerações finais sobre o andamento do projeto e indica direções para a continuação do estudo na segunda etapa do Trabalho de Conclusão de Curso.

2 Revisão Bibliográfica

2.1 Fundamentação Teórica

Segundo (Lessa, Demian, 1999), “Estatisticamente, enquanto 30% dos custos de uma infraestrutura computacional estão diretamente associados à aquisição de hardware, os 70% restantes dizem respeito à manutenção e suporte aos recursos e serviços nela contidos.” Além disso, (STALLINGS, 1998), afirma que “redes de computadores que não possuem gerenciamento estão sujeitas a não continuar funcionando por muito tempo”. Diante desse cenário, o gerenciamento da infraestrutura de TI torna-se uma atividade estratégica e indispensável para garantir a continuidade e eficiência dos serviços prestados por uma organização.

O gerenciamento e o monitoramento contínuos permitem que os administradores tenham uma visão abrangente, estruturada e em tempo real do estado da rede, o que viabiliza a detecção precoce de anomalias, a prevenção de falhas e a tomada de decisões mais assertivas.

2.1.1 SRE

A Engenharia de Confiabilidade de Sites (*Site Reliability Engineering* – SRE) é uma metodologia criada originalmente pelo Google no início dos anos 2000, com a proposta de aplicar fundamentos da engenharia de software na administração de operações e infraestrutura. Em 2003, o conceito foi estruturado por Ben Treynor Sloss, engenheiro da empresa, que descreveu o SRE como "o resultado de pedir a um engenheiro de software para construir uma equipe de operações"(BEYER C. JONES, 2016). Diferentemente dos métodos convencionais, o SRE traz uma abordagem orientada por métricas precisas, alto grau de automação e foco constante na estabilidade e desempenho dos sistemas.

Apesar de ter sido inicialmente associado à operação de sistemas de grande escala, os princípios do SRE são plenamente adaptáveis à administração da infraestrutura de TI. Ao considerar a infraestrutura como um serviço essencial que deve ser continuamente monitorado e aprimorado, as equipes podem adotar práticas de SRE para aumentar a resiliência de componentes como redes, servidores, sistemas de backup e soluções de armazenamento.

Na prática, isso envolve estabelecer Indicadores de Nível de Serviço (SLIs) e Objetivos de Nível de Serviço (SLOs) voltados para recursos da infraestrutura, como disponibilidade de conexões, tempo de resposta da rede, sucesso em procedimentos de backup, tempo médio de resolução de falhas, entre outros. Além disso, a automação de tarefas operacionais, a adoção de análises pós-incidente sem responsabilização individual e o uso de monitoramento constante são estratégias que contribuem para a robustez de ambientes físicos e virtuais.

O modelo SRE baseia-se em pilares técnicos e culturais que norteiam sua aplicação:

Observabilidade: Observabilidade pode ser definida como a capacidade de inferir e compreender o estado interno de um sistema complexo a partir da análise de suas saídas, principalmente por meio da coleta e correlação de métricas, *logs* e *traces*.

SLIs (*Service Level Indicators*): Métricas técnicas e objetivas utilizadas para mensurar a qualidade do serviço prestado. Exemplos incluem tempo médio de resposta, taxa de erros, latência de rede e disponibilidade percentual.

SLOs (*Service Level Objectives*): Metas específicas de qualidade do serviço associadas aos SLIs. Essas metas estabelecem limites aceitáveis para métricas críticas. Por exemplo, estabelecer que a latência de uma API deve permanecer abaixo de 100 ms em 99% das requisições.

***Error Budget*:**Margem de erro tolerável dentro do SLO, ou seja, a quantidade máxima aceitável de falhas ou degradação do serviço dentro de um intervalo de tempo, sem consequências contratuais.

Eliminação de Toil: Refere-se à redução ou automação de tarefas manuais, repetitivas e de baixo valor agregado que consomem tempo da equipe. Esse tipo de atividade, como reinícios manuais de serviços, ajustes frequentes em configurações ou execução de scripts rotineiros, não contribui diretamente para a escalabilidade ou melhoria do sistema, além de aumentar o risco de erros humanos.

Postmortem: pós incidentes, realiza-se uma análise retrospectiva documentada, que busca identificar causas-raiz, oportunidades de melhoria e planos de ação, sem apontar culpados, promovendo uma cultura de aprendizado contínuo.

Esses elementos tornam o SRE uma prática orientada a dados, onde decisões técnicas são fundamentadas em métricas objetivas e não em percepções subjetivas.

2.1.2 Observabilidade

O termo observabilidade nasceu em agosto de 1960 através de Rudolf E. Kálmán em seu livro *On the General Theory of Control Systems)* (KáLMáN, 1960) definindo como a medida dos estados internos de um sistema podem ser captados através do conhecimento das saídas externas. Em outras palavras, é uma medida que descreve quão bem podem os estados de um sistema serem inferidos a partir do conhecimento dos dados de entrada e saída.

A observabilidade concentra-se em externar o máximo de dados possível sobre todo o serviço, permitindo aos profissionais inferir sobre o estado do sistema. A observabilidade é composta por três pilares: métricas, *logs* e *traces*.

Métricas são a base do monitoramento e expressam parte do estado interno de um sistema, como por exemplo, consumo de memória, taxa de escrita e leitura em disco, entre outras. Normalmente são definidas ao longo de um período de tempo e são muito úteis na detecção e alertas. Geralmente, as métricas são coletadas por ferramentas especializadas, como o Zabbix e Prometheus. Essas ferramentas oferecem templates pré-configurados para a coleta de dados e permitem a configuração de métricas personalizadas conforme as necessidades específicas do sistema. Além disso, é possível configurar alarmes que notificam os administradores sobre quaisquer desvios ou problemas detectados.

Logs são registros detalhados de eventos que ocorrem dentro de um sistema. Geralmente são em forma de texto. Os logs ajudam a aprofundar e entender o problema. Porém pode ser desafiador identificar e até mesmo realizar uma análise correta do registro. O log excessivo pode ser um problema, por outro lado, o log insuficiente também pode prejudicar a identificação de eventos.

Traces descrevem toda a jornada de uma solicitação e sua movimentação. Ajudam a entender onde está acontecendo o problema. São úteis para solucionar problemas de dependências de serviços, identificar atrasos, gargalos e entender o desempenho geral de um sistema.

A combinação desses três elementos possibilita uma visão abrangente e detalhada do sistema, permitindo que equipes de infraestrutura atuem de forma mais eficaz na identificação e prevenção de falhas.

2.1.3 Monitoramento

De acordo com o Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) (IEEE, 1990), o monitoramento é definido como uma ferramenta de software ou dispositivo de hardware que opera simultaneamente com um sistema ou componente e supervisiona, registra, analisa e verifica a operação do sistema ou componente. Em outras palavras, é o processo de coletar, analisar e usar dados sobre a integridade e desempenho de um ambiente.

O objetivo principal do monitoramento é detectar desvios e anomalias, alertando os profissionais responsáveis quando os valores excedem os limites pré-definidos, indicando potenciais problemas. Este modelo funciona muito bem para problemas conhecidos. Contudo, pode ser bastante falho em cenários onde a imprevisibilidade é alta.

Uma característica marcante no monitoramento é a geração de dados quantitativos. Pois geralmente, o monitoramento está ligado à coleta de métricas específicas como uso de CPU, memória, largura de banda, entre outras. Essas métricas devem ser revisadas constantemente e, caso necessário, devem ser recalibradas. Assim, evitar a fadiga de alertas.

Segundo (BEYER C. JONES, 2016), existem quatro sinais de ouro para o monitoramento: latência, tráfego, falhas e saturação. Latência representa o tempo de resposta de uma requisição,

por exemplo, o tempo de resposta de um comando *ping* medido em milissegundos. Tráfego é a quantidade de demanda que está sendo alocada para um sistema a partir de uma métrica específica. Por exemplo, para a largura de banda, pode ser o *download* e *upload* medidos em megabits por segundo. Para um serviço *web*, pode ser a taxa de requisições HTTP respondidas por segundo. As falhas expressam métricas de taxas explícitas de erro, como, por exemplo, requisições HTTP que retornam erro 500. Podem ser implícitas, por exemplo, resposta HTTP 200, porém associada a um conteúdo incorreto. Ou ainda por políticas, por exemplo, toda resposta HTTP 200 que excedeu 1 segundo. E por fim, a saturação, que corresponde à carga suportada por um sistema, como a taxa de operações de entrada/saída em um banco de dados ou o número de dispositivos conectados a um ponto de acesso (*Access Point*).

2.1.4 Monitoramento x Observabilidade

Muitos profissionais ainda se confundem quanto às diferenças entre monitoramento e observabilidade. De acordo com (USMAN, 2022), o monitoramento é o processo de acompanhar a saúde de um sistema por meio de um conjunto predefinido de métricas e *logs*. Já a observabilidade permite descobrir e entender comportamentos inesperados, ajudando na identificação de problemas que não foram previstos previamente.

Segundo (MUNIZ A. ARANHA, 2024), o monitoramento tem como objetivo medir o funcionamento de sistemas, geralmente baseado em regras de notificação que alertam para possíveis anomalias em serviços. As métricas e os *logs* monitorados devem ser bem definidos e revisados periodicamente, a fim de manter sua eficácia. Por outro lado, a observabilidade aborda um conceito mais amplo que se refere à capacidade de entender o estado interno de um sistema com base em seus *outputs* externos.

Segundo (MUNIZ A. ARANHA, 2024), a diferença básica entre observabilidade e monitoramento é que o monitoramento é reativo e a observabilidade é proativa, ou seja, enquanto o monitoramento pode identificar o problema, a observabilidade antecipa aqueles que podem acontecer.

A observabilidade trabalha de forma complementar ao monitoramento. Enquanto o monitoramento foca em métricas predefinidas, a observabilidade envolve a coleta e a análise de *logs* e métricas para obter uma visão completa do comportamento do sistema.

2.1.5 Níveis de Serviços

Os níveis de serviços são conceitos essenciais para a gestão de confiabilidade e desempenho em sistemas e infraestrutura. Eles ajudam a definir regras claras sobre o desempenho e entrega de serviços, garantindo que as operações sejam acompanhadas por métricas bem definidas e objetivos alcançáveis, além de acordos pré-definidos.

Em (BEYER C. JONES, 2016) os autores explicam que os níveis de serviços são adquiridos pela intuição, experiência e a compreensão do que os usuários desejam. Essas medidas descrevem as propriedades básicas das métricas que importam, quais valores queremos que essas métricas tenham e como reagiremos se não pudermos fornecer o serviço esperado. Em última análise, a escolha de métricas adequadas ajuda a impulsionar a ação certa se algo der errado e também dar à equipe de infraestrutura a confiança de que um serviço é saudável. Existem 3 níveis de serviços: SLA, SLO e SLI.

SLA *Service Level Agreement* ou Acordo de Nível de Serviço é um contrato formal entre o provedor de serviço e seus clientes que define as expectativas em relação à qualidade e à disponibilidade do serviço. Esses contratos especificam consequências caso essas expectativas não sejam atendidas, como compensações ou multas. O SLA é voltado para um contexto mais comercial e pode envolver aspectos como: tempo máximo de resposta a incidentes, percentual de disponibilidade do serviço (*uptime*), entre outros.

SLO *Service Level Objective* ou Objetivo de Nível de Serviço, é uma meta técnica que o provedor de serviços se compromete a atingir. O SLO serve para alinhar as expectativas de negócio com a operação técnica, em outras palavras, é o alvo acordado para o nível de serviço. O SLO, por exemplo, pode ser: "A disponibilidade de internet para o sistema X deve ser de 99,0%".

SLI *Service Level Indicator* ou Indicador de Nível de Serviço é a métrica específica usada para medir o desempenho de um serviço em relação ao SLO definido. Trata-se de um indicador quantitativo que reflete o estado real do serviço. Exemplos comuns de SLIs incluem taxa de erro, tempo de resposta (latência), disponibilidade e vazão (*throughput*). A comparação entre os SLIs medidos e os SLOs definidos permite avaliar se o serviço está dentro dos padrões estabelecidos.

$$\text{SLI} = \left(\frac{\text{good events}}{\text{valid events}} \right) \times 100$$

Figura 2.1 – Cálculo do SLI.
Fonte: (CLIMENT, 2020)

A definição clara de SLAs, SLOs e SLIs permite às equipes técnicas agir com maior precisão diante de falhas, além de oferecer uma base objetiva para negociações e melhorias contínuas no ambiente de TI.

2.1.6 Error Budget

O conceito de *Error Budget* (ou orçamento de erro) é a tolerância máxima de tempo de indisponibilidade para um serviço em um período específico sem consequências contratuais. Esse conceito está diretamente relacionado ao SLO, que define o nível de confiabilidade esperado para um serviço. Por exemplo, se o SLO estipula que uma aplicação deve estar disponível em 99% do tempo durante um mês, o *Error Budget* corresponderá ao 1% restante, ou seja, o tempo que pode ser "gasto" em falhas, manutenções planejadas ou mesmo em incidentes imprevistos.

O valor do *Error Budget* não deve ser interpretado apenas como uma margem de tolerância para falhas, mas sim como um instrumento de tomada de decisão. Quando o consumo do orçamento se encontra dentro dos limites definidos, a equipe pode avançar com novas implementações ou mudanças mais arriscadas. Por outro lado, quando o orçamento está esgotado ou prestes a ser consumido, deve-se priorizar a estabilização do sistema, reduzindo a introdução de novas funcionalidades e focando em ações de confiabilidade.

2.1.7 Eliminação de TOIL

Segundo (BEYER C. JONES, 2016) TOIL *(Tasks Often Involving Legwork)* "é o tipo de trabalho vinculado a execução de um serviço de produção que tende a ser manual, repetitivo, automatizável, tático, desprovido de valor duradouro e que se dimensiona linearmente à medida que o serviço cresce". Na prática, nem todas as tarefas que se encaixam na descrição original apresentam todos os atributos citados. Entretanto, (BEYER C. JONES, 2016) diz que a chave para identificar o TOIL em uma tarefa é reconhecer a presença de pelo menos um dos seguintes atributos:

Manual: tarefa que exigem interação humana direta e repetitiva, sem automação possível.

Repetitivo: tarefas que seguem um padrão fixo, sem variação significativa.

Automatizável: tarefas com o potencial para serem automatizadas, mas que ainda são realizadas manualmente por falta de implementação ou conhecimento técnico.

Não Tático: tarefas com foco em ações imediatas e sem impacto estratégico a longo prazo.

Desprovido de valor duradouro: tarefas que não geram benefícios a longo para a empresa ou para o usuário final.

Dimensionamento linear:tarefas que aumentam em proporção direta ao crescimento do serviço, demandando mais tempo e recursos humanos sem otimização.

2.1.8 Postmortem

Postmortem é uma prática estruturada de análise e documentação de incidentes que ocorreram em sistemas de produção. Seu principal objetivo é identificar as causas-raiz de falhas, avaliar o impacto gerado e propor medidas corretivas e preventivas, de forma a reduzir a recorrência de problemas semelhantes.

Um *Postmortem* não se limita a descrever o que aconteceu, mas busca detalhar quando, como e por que o incidente ocorreu, bem como quais ações imediatas foram tomadas para mitigar seus efeitos. Além disso, recomenda-se que este processo seja conduzido sob a abordagem de *blameless postmortem* (análise sem culpabilização), a fim de fomentar uma cultura organizacional voltada ao aprendizado coletivo e à melhoria contínua, em vez de punir indivíduos.

A prática de *Postmortem* gera benefícios relevantes, como o aumento da transparência entre equipes técnicas e áreas de negócio, a melhoria da confiabilidade do sistema e a criação de um repositório de conhecimento que pode ser consultado em futuros incidentes.

2.1.9 Maturidade da rede

(MCGILLICUDDY, 2022) propõe um modelo composto por quatro níveis para medir a maturidade da observabilidade em redes. O primeiro nível, denominado Gerenciamento Reativo, é caracterizado pela atuação pontual e emergencial das equipes, que respondem a incidentes de forma isolada, sem estratégias preventivas bem definidas.

O segundo nível, chamado Gerenciamento Proativo, as equipes de infraestrutura utilizam ferramentas centralizadas de monitoramento e observabilidade, o que permite maior visibilidade do ambiente e melhora a capacidade de resposta a incidentes.

No terceiro nível, o Gerenciamento Orientado a Serviços, as equipes passam a operar como prestadoras de serviços. Nesse contexto, já existem contratos (SLAs) bem definidos, e os dados coletados são utilizados para apoiar a tomada de decisões e melhorar a qualidade dos serviços oferecidos.

O quarto nível é o Gerenciamento Dinâmico. Nesse nível, além da adoção de ferramentas e processos automatizados para a resolução de incidentes, são aplicadas análises avançadas de todo o ambiente de rede. Essa abordagem permite uma visão holística da infraestrutura, otimização contínua dos recursos e maior capacidade de adaptação frente a mudanças nos requisitos ou no contexto operacional.

2.1.10 SNMP

O protocolo SNMP *Simple Network Management Protocol* é um protocolo de gerência definido a nível de aplicação que foi desenvolvido na década de 1980 pelo IETF *Internet Engineering Task Force* com o objetivo de simplificar o gerenciamento de equipamentos conectados à

rede. O SNMP segue o modelo Gerente-Agente 2.2. Os gerentes, que podem ser um ou vários, contêm a aplicação de monitoramento e normalmente são instalados em servidores dedicados a esta operação de gestão. Já os agentes são processos que rodam nos dispositivos gerenciados.

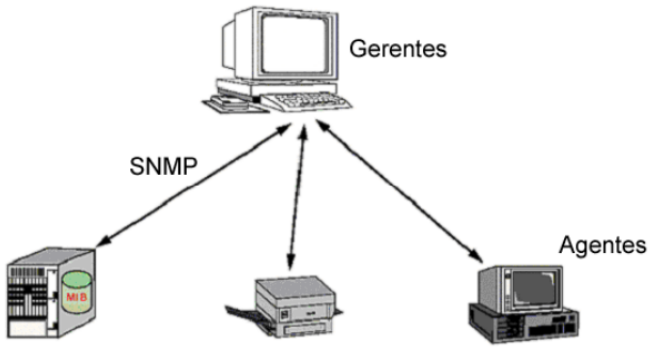


Figura 2.2 – Arquitetura SNMP.
Fonte: <https://cbpfindex.cbpf.br/publication_pdfs/nt00601.2010_10_15_11_40_12.pdf>

Atualmente, existem três versões principais do protocolo: SNMPv1, SNMPv2c e SNMPv3. O SNMPv1 é a versão original, com funcionalidades básicas, como os comandos *get*, *set* e *trap*. A versão SNMPv2c introduziu melhorias na comunicação, como comandos em lote e respostas mais detalhadas. Já o SNMPv3 acrescenta mecanismos de segurança mais robustos, incluindo autenticação e criptografia, o que o torna mais adequado para ambientes críticos, distribuídos e mais aderentes aos sistemas atuais.

Evolution of SNMP Standard

SNMP Version		RFC	Highlights
SNMPv1		1155, 1157, 1212	First SNMP implementation, used private and public community strings for security. Five operations: Get, Set, GetNext, Response, and Trap
SNMPv2	SNMPv2	1441–1452	Major improvements in performance with "GetBulk" and better security
	SNMPv2u	1909, 1910	Easier configuration
	SNMPv2*	IETF Draft	Remote configuration
	SNMPv2c	1901–1908	No remote configuration, uses community strings
SNMPv3		2271–2275	Takes most of the improvements from SNMPv2. Adds strong security and authentication model, support for remote agent configuration with SNMP, and unique ID for each SNMP engine

Figura 2.3 – Evolução do protocolo SNMP.
Fonte: (PHALTANKAR, 1999)

2.1.11 Management Information Base (MIB) e Object Identifier (OID)

A *Management Information Base* (MIB), como o próprio nome sugere, é uma base de dados utilizada para o gerenciamento de informações de dispositivos em rede. Essa base é organizada hierarquicamente em uma estrutura de árvore, na qual cada entrada é identificada por um *Object Identifier* (OID), um identificador único que aponta para uma informação específica.

A MIB atua como uma espécie de tradutor entre o gerenciador de rede e os dispositivos monitorados, permitindo operações de consulta, monitoramento, geração de alertas e envio de comandos por meio do protocolo SNMP. Cada dispositivo que implementa o SNMP deve seguir a estrutura básica padronizada da MIB, definida pelo IETF e documentada por meio de RFCs (*Request for Comments*) e os fabricantes de equipamentos podem estender a MIB com ramos privados que permitem monitorar funcionalidades específicas de seus dispositivos.

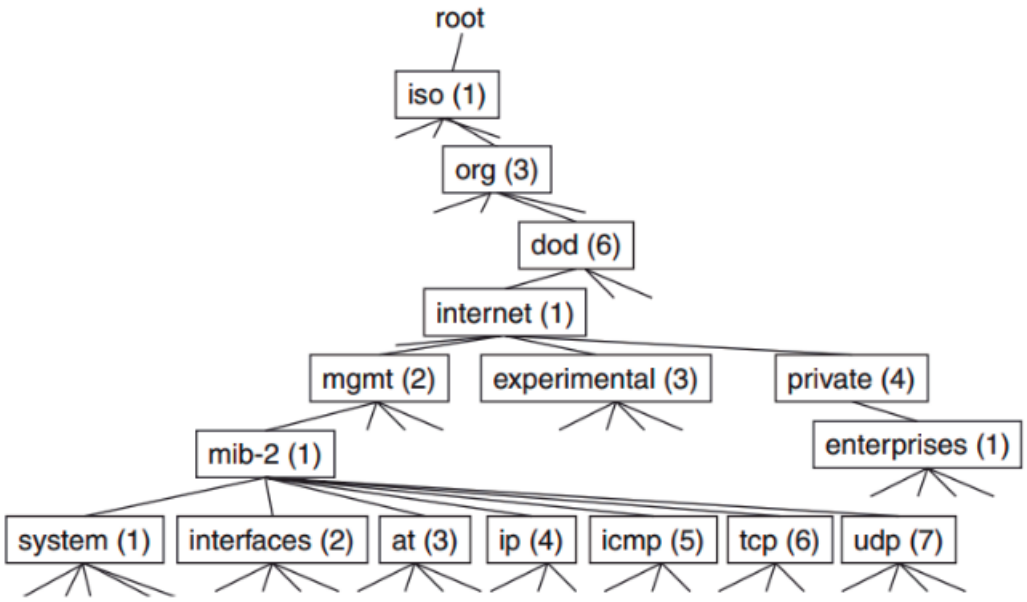


Figura 2.4 – Estrutura MIB.
Fonte:(CLEMM, 2006)

Resumidamente, a MIB combina padronização e flexibilidade no gerenciamento de sistemas computacionais atuais, o que a torna uma ferramenta essencial para redes heterogêneas, ao prover recursos tanto de observabilidade quanto de monitoramento.

2.1.12 Zabbix

O Zabbix (Zabbix LLC, 2025) é um software de código aberto desenvolvido para ser uma solução completa de monitoramento. Desenvolvido para ser uma solução completa, esse software permite coletar e armazenar dados, enviar alertas, gerar relatórios de disponibilidade, realizar descobertas automáticas de dispositivos e serviços, entre outras funcionalidades.

O Zabbix possui várias ferramentas para realizar o monitoramento dos dispositivos. As coletas podem ser feitas por SNMP, IPMI, *scripts* personalizados, e várias outras formas além dos agentes. Também é possível coletar métricas de várias fontes como arquivos de *log*, base de dados, sensores IoT, APIs, *end points* HTTP/HTTPS demonstrando flexibilidade e capacidade de adaptação a diferentes ambientes.

O Zabbix mantém uma comunidade bastante ativa onde é bem comum encontrar fóruns de discussões sobre dúvidas da ferramenta. Da mesma forma, possui documentação bastante completa e em português. Além disso, é possível encontrar *templates* prontos para os dispositivos mais comuns, contendo itens, gatilhos (*triggers*), gráficos e regras de descoberta, o que reduz significativamente o tempo de configuração.

Dada sua robustez e capacidade de integração, o Zabbix é uma das principais opções para equipes de infraestrutura que buscam monitoramento centralizado e escalável de ambientes complexos (Zabbix LLC, 2025).

2.1.13 Graylog

Segundo (STALLINGS, 2016), "*logs são arquivos que mantêm registros detalhados de eventos e transações, e servem como base para a análise de incidentes, manutenção preventiva e rastreamento de ações executadas em um sistema computacional.*"

Devido à variedade e quantidade de sistemas, aplicações e dispositivos que geram *logs*, é de suma importância ter um centralizador de *logs* configurado na rede. O Graylog (Graylog, Inc., 2025) é uma ferramenta de código aberto desenvolvida com esse propósito, oferecendo uma solução eficiente para a gestão centralizada de *logs* e que possui alta compatibilidade com sistemas operacionais, aplicações e serviços diversos. Graylog permite a recepção de *logs* por meio de múltiplos protocolos e formatos, como Syslog, GELF, Beats, JSON, entre outros. Além disso, a plataforma oferece recursos de busca, filtragem e visualização em tempo real, o que facilita a identificação de padrões, falhas e atividades suspeitas no ambiente monitorado.

Além dessas características, o Graylog possui uma arquitetura modular e escalável, permitindo sua adaptação a ambientes corporativos de diferentes portes.

2.1.14 Grafana

O Grafana (Grafana Labs, 2025) é uma ferramenta *open source* desenvolvida com foco na visualização e análise de dados em tempo real. Tem como objetivo transformar dados brutos em *dashboards* interativos e altamente configuráveis. Ele é capaz de se conectar a diversas fontes de dados, como bancos de dados, sistemas de monitoramento e ferramentas de coleta de métricas.

A principal característica do Grafana é sua interface gráfica intuitiva, que facilita a construção de *dashboards*. Além disso, o Grafana permite a adição de *plugins* que expandem suas

funcionalidades, incluindo novos tipos de visualizações, temas e integrações com ferramentas de terceiros.

Embora não utilize uma linguagem única de consulta, o Grafana adapta-se à linguagem nativa de cada fonte de dados, o que o torna versátil, mas exige que o usuário compreenda as particularidades de cada fonte para construir consultas e visualizações eficientes.

3 Trabalhos Relacionados

Esta seção apresenta trabalhos relacionados à adoção de práticas de SRE, bem como a adaptação desses princípios em outro contexto. Destaca estudos que envolvem a implantação e avaliação das ferramentas de monitoramento utilizadas no presente projeto. Além disso, são discutidas propostas de planos de ação para a continuidade dos serviços em caso de falhas no sistema de monitoramento, bem como a análise de um estudo que propõe a previsão de indicadores de desempenho em tempo real.

A pesquisa de (AZAMBUJA, 2023) tem como foco principal o aprimoramento do sistema de monitoramento da infraestrutura de TI do Hospital Universitário de Santa Maria (HUSM), apresentando grande semelhança com o presente trabalho. Para isso, os autores analisaram as ferramentas existentes — Zabbix e Cacti — e propuseram uma nova lógica de monitoramento, baseada em *templates* personalizados e na automatização da descoberta e cadastro de dispositivos. Diferentemente da abordagem adotada neste estudo, no trabalho de (AZAMBUJA, 2023), o Zabbix foi utilizado exclusivamente para alertas e incidentes em tempo real, enquanto o Cacti ficou responsável pela geração de gráficos e retenção histórica de dados.

O levantamento realizado pelos autores também identificou problemas como uso excessivo de recursos computacionais, coletas redundantes e cadastros desatualizados nas ferramentas de monitoramento. As mudanças implementadas promoveram um monitoramento mais eficiente da infraestrutura de rede e otimização da automação dos processos, reduzindo a necessidade de intervenção humana.

O artigo (DAVE, 2023), investiga como práticas de SRE, originalmente concebidas no Google para o gerenciamento de operações em ambientes de desenvolvimento de *software*, vêm sendo adaptadas para setores industriais, visando a melhoria da eficiência operacional e a redução do *downtime* em ambientes de manufatura. O autor demonstra que, além do contexto do Google, outras organizações industriais passaram a explorar os benefícios do SRE, aplicando suas práticas à automação de processos físicos, resposta rápida a incidentes e aumento da confiabilidade de sistemas produtivos. Destacam-se, no estudo, casos práticos de aplicação de metodologias SRE associadas a ferramentas de monitoramento e automação alinhadas às tendências da transformação digital no setor manufatureiro.

Em relação ao presente trabalho, que propõe a aplicação das práticas de SRE especificamente no contexto da infraestrutura de TI, observa-se uma similaridade na intenção de adaptar as práticas originalmente voltadas ao desenvolvimento/operações de *software* para novas áreas técnicas. Enquanto o artigo de (DAVE, 2023) enfoca o uso do SRE na modernização de ambientes industriais físicos, este trabalho concentra-se na aplicação de princípios como definição de monitoramento, níveis de serviço, eliminação de TOIL e alertas para aprimorar a observabilidade

e resiliência da infraestrutura tecnológica (redes, servidores e equipamentos de comunicação). Ambos os estudos reforçam a versatilidade do SRE fora do escopo tradicional do desenvolvimento de *software*, apontando seu potencial estratégico na melhoria de ambientes operacionais críticos.

O trabalho de (SOUZA, 2024), apresenta uma análise aprofundada sobre a metodologia SRE e sua aplicabilidade em ambientes corporativos. O autor explora desde os conceitos fundamentais do SRE, destacando a integração entre engenharia de *software* e operações, até a aplicação prática nas organizações, abordando aspectos como automação, eliminação do TOIL, uso de SLIs, SLOs e SLAs, além da importância da observabilidade. O estudo também discute as principais responsabilidades de um engenheiro SRE e a necessidade de estrutura organizacional e treinamentos adequados para garantir uma implementação bem-sucedida da metodologia.

Este trabalho contribui para o tema ao demonstrar como a adoção do SRE pode aumentar a confiabilidade, escalabilidade e eficiência dos serviços de TI, bem como reduzir falhas nos sistemas.

Por sua vez, (SILVA, 2024) abordam o monitoramento de equipamentos de terceiros, adotando conexões VPN *(Virtual Private Network)* para a troca segura de informações entre os agentes e o gerente. Embora essa solução tenha proporcionado segurança na comunicação entre redes distintas, sua operação revelou-se complexa: manter VPNs ativas, monitoradas e seguras para cada cliente exige esforços consideráveis e pode gerar atrasos ou falsos alertas em caso de latência ou queda da conexão. Como alternativa, é sugerida a utilização de servidores *proxy* do Zabbix em cada cliente, permitindo a coleta contínua de dados mesmo durante falhas de comunicação. Assim como neste trabalho, (SILVA, 2024) também implementaram *dashboards* no Grafana, possibilitando uma visualização rápida da saúde dos equipamentos monitorados e análises detalhadas.

Este trabalho contribui para o tema ao identificar potenciais falhas de comunicação entre a ferramenta de monitoramento e clientes externos, que podem resultar em registros de falso-positivos.

No contexto da Governança de TI, (AGUIAR, 2013) propõem a implantação da ferramenta Zabbix no CCA-RJ (Centro de Computação da Aeronáutica do Rio de Janeiro), alinhando o monitoramento às boas práticas previstas no modelo COBIT. O trabalho enfatiza que o sucesso do monitoramento depende não apenas da solução técnica escolhida, mas também da adoção de práticas organizacionais, como treinamento, definição de processos e alinhamento estratégico. A contribuição de (AGUIAR, 2013) complementa a presente pesquisa ao reforçar a importância da integração entre práticas técnicas e organizacionais, destacando que soluções de monitoramento devem também apoiar a continuidade dos serviços e a resiliência da infraestrutura.

Já (SILVA, 2021) apresenta uma proposta de elaboração de um PCN (Plano de Continuidade de Negócios) voltado para o ambiente de TI de um hospital público de grande porte em Santa Catarina. O trabalho parte do entendimento de que a área de TI é essencial para a manutenção dos

processos hospitalares e que a indisponibilidade de sistemas críticos, como o ERP, servidores de banco de dados e impressoras, pode comprometer a qualidade e a agilidade do atendimento aos pacientes. Para mitigar esses riscos, o autor propõe um conjunto estruturado de ações baseadas em normas e boas práticas de gestão de continuidade, como a realização de análise de impacto nos negócios e a gestão de riscos conforme a norma. (SILVA, 2021) complementa este trabalho na elaboração de um plano de continuidade ao estruturar diretrizes de ação e recuperação em eventos disruptivos.

No que se refere à segurança da rede, (FUZI, 2022) apresentam uma integração do Zabbix com o serviço de mensagens Telegram, visando à detecção e comunicação rápida de ataques de negação de serviço (DoS), como *ping flooding* e *SYN flooding*. O objetivo do projeto foi automatizar a detecção desses ataques e alertar imediatamente os administradores de rede. A pesquisa também utilizou ferramentas auxiliares, como *tcpdump* e *netstat*, para reforçar a precisão na detecção de anomalias no tráfego de rede.

Além disso, o trabalho de (FUZI, 2022) descreve o desenvolvimento e a validação de um sistema de monitoramento capaz de enviar alertas automáticos para os administradores ao identificar possíveis ataques, demonstrando a importância da comunicação rápida para mitigar ameaças.

Por fim (PEIXIAN, 2020) propõe uma plataforma de monitoramento e predição em tempo real para *clusters*, voltada para usuários corporativos e baseada no Zabbix. A plataforma monitora continuamente diversos indicadores do *cluster* e prevê indicadores de desempenho importantes em tempo real. Ao detectar ou prever situações anormais, o sistema alerta os usuários, facilitando a compreensão imediata da operação do *cluster* e permitindo ações preventivas ou corretivas mais rápidas. O trabalho detalha o modelo de predição de curto prazo em tempo real utilizando o método da média móvel simples SMA *(Simple Moving Average)*. O SMA é aplicado aos dados em uma janela deslizante a fim de eliminar flutuações irregulares e revelar a tendência de longo prazo da série temporal da métrica monitorada. Com base na saída da médica móvel, (PEIXIAN, 2020) aplica o algoritmo TAR*(Trend-Aware Regression)* para prever, em curto prazo e tempo real, o comportamento futuro de indicadores de desempenho importantes.

4 Desenvolvimento

4.1 Metodologia Científica

Este estudo caracteriza-se como uma pesquisa aplicada, com abordagem experimental, pois busca avaliar, em um ambiente real de infraestrutura de TI, os impactos da integração de práticas de SRE na resiliência da rede. A pesquisa possui caráter qualitativo e quantitativo. Qualitativo, por analisar comportamentos e padrões de falhas na rede e quantitativo por coletar métricas de disponibilidade, latência, erros e incidentes registrados antes e depois da aplicação do modelo proposto.

A metodologia adotada neste trabalho visa aplicar as práticas de SRE adaptadas ao contexto de uma infraestrutura de rede corporativa de médio porte, com o objetivo de aprimorar a observabilidade, detectar incidentes de forma proativa e reduzir o tempo de resposta a falhas. Para isso, foram definidas cinco etapas:

1. Mapeamento do ambiente
2. Definição de métricas, SLOs e SLIs
3. Instrumentação do ambiente
4. Coleta de Dados
5. Eliminação de TOIL

4.1.1 Mapeamento do ambiente

Inicialmente, foi realizado o levantamento dos recursos da infraestrutura de rede e a identificação dos serviços essenciais. Para esse mapeamento, foi consultada a documentação existente da infraestrutura, que se encontrava desatualizada. Além disso, foram realizadas consultas ao sistema de *Service Desk* com o objetivo de identificar pontos de falha não formalmente mapeados.

Nesta etapa, foi possível identificar a arquitetura da rede corporativa, composta pela matriz e duas filiais. Cada unidade conta com dois links de conexão à internet: um Link Principal e um Link Secundário. O Link Principal é fornecido pelo mesmo provedor de serviços para todas as unidades, que, além da conexão com a internet, também disponibiliza uma VPN corporativa entre a matriz e as filiais. Essa VPN é gerenciada pelo próprio provedor, garantindo uma comunicação segura entre as unidades. Além disso, o provedor de internet principal fornece e gerencia o sistema de telefonia fixa da empresa.

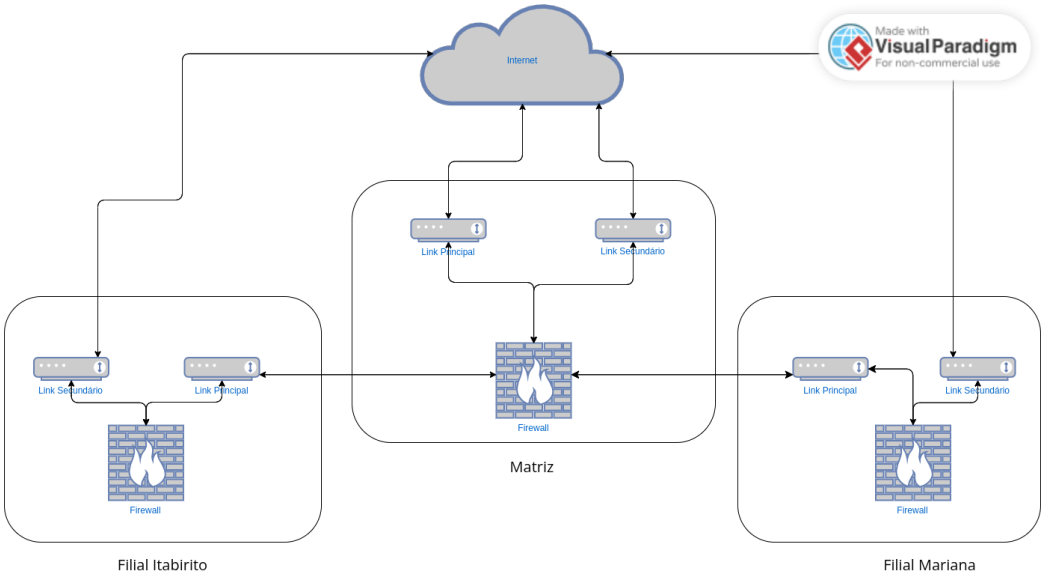


Figura 4.1 – Topologia de rede.
Fonte: Autoria própria

No entanto, foi observado que a matriz centraliza a única porta de saída para a internet quando as filiais estão utilizando o link principal. Ou seja, todas as conexões externas originadas nas filiais trafegam inicialmente pela matriz antes de serem roteadas para a internet. Essa característica evidencia um ponto de falha crítico: caso ocorra indisponibilidade no link da matriz, as filiais também perdem a capacidade de comunicação externa, impactando serviços essenciais como o sistema de telefonia.

Para mitigar esse risco, cada unidade conta com um Link Secundário. Em caso de falha no Link Principal, o *firewall* local de cada unidade realiza automaticamente a troca para o Link Secundário.

Abaixo do *firewall*, a topologia é composta por *switches core*, responsáveis pelo roteamento interno e pela comunicação entre os servidores do *cluster*, e por *switches* de distribuição, que interligam os segmentos da rede corporativa. Esses equipamentos conectam os diversos ativos da infraestrutura, incluindo *access points*, estações de trabalho, telefones, impressoras e demais dispositivos essenciais ao funcionamento da rede.

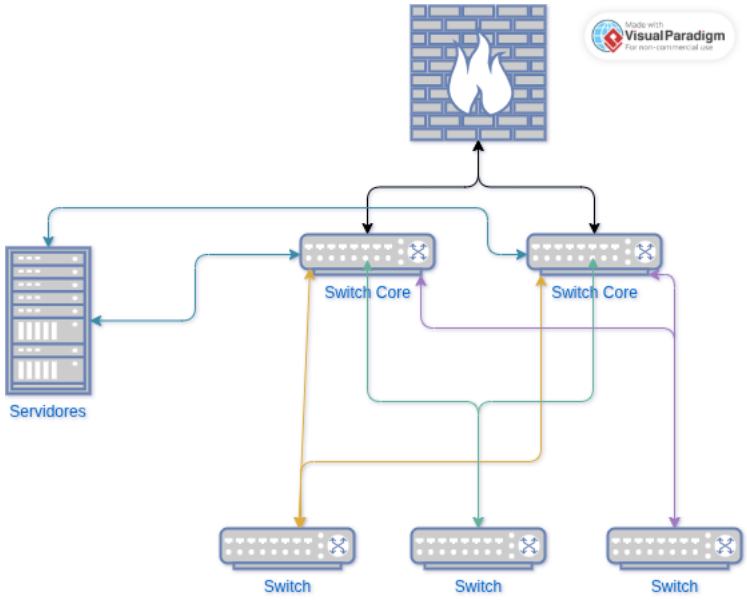


Figura 4.2 – Topologia interna.
Fonte: Autoria própria

4.1.2 Definição de métricas, SLOs e SLIs

Após o mapeamento da infraestrutura de rede, foi realizada a definição das métricas de monitoramento, juntamente com os SLOs e SLIs para cada grupo de dispositivos e serviços críticos.

A definição dos SLIs e SLOs considerou tanto o grau de importância de cada serviço quanto o nível de investimento destinado à sua confiabilidade. A seguir, apresentam-se as métricas e os objetivos estabelecidos para cada grupo de elementos da topologia:

1. **Links de Internet (Primário e Secundário)**
   - **Métricas:** Latência, largura de banda utilizada (*upload* e *download*), disponibilidade do link e taxa de perda de pacotes.
   - **SLIs:** Disponibilidade do link em %, latência média e taxa de perda de pacotes.
   - **SLOs propostos:**
     - Disponibilidade $\geq$ 99% para os links;
     - Latência média $\leq$ 60 ms para destinos críticos (*SaaS*, comunicação entre matriz e filiais);
     - Perda de pacotes $\leq$ 1%.
2. **Firewall**
   - **Métricas:** Uso de CPU e memória, quantidade de sessões ativas e conexões simultâneas.

- **SLIs:** Taxa de utilização de CPU e memória, número de sessões ativas em relação à capacidade máxima.
- **SLOs propostos:**
  - Uso médio de CPU e memória $\leq 70\%$;
  - Disponibilidade $\geq 99\%$;

3. **Switches (Core e Distribuição)**

- **Métricas:** Estado das interfaces e portas críticas, taxa de utilização dos *uplinks*, erros e descartes de pacotes por porta e status de energia.
- **SLIs:** Disponibilidade das portas principais, taxa de erros por interface e saturação de *uplinks* ($< 80\%$).
- **SLOs propostos:**
  - Disponibilidade $\geq 99\%$ para *switches* (core e distribuição);
  - Taxa de erros e descartes $< 0{,}1\%$ do tráfego;
  - *Uplinks* abaixo de 80% da capacidade.

4. **Access Points (APs)**

- **Métricas:** Número de clientes conectados, força e qualidade do sinal, largura de banda utilizada.
- **SLIs:** Disponibilidade do AP.
- **SLOs propostos:**
  - Disponibilidade $\geq 98\%$;
  - Ocupação de até 80% da capacidade do AP.

5. **Servidores Físicos e *Hypervisors***

- **Métricas:** Uso de CPU, memória e armazenamento, status de interfaces de rede, temperatura, alertas de *hardware* e disponibilidade das VMs (Virtual Machines) hospedadas.
- **SLIs:** Disponibilidade dos servidores, utilização de recursos e alertas de hardware.
- **SLOs propostos:**
  - Disponibilidade $\geq 99\%$;
  - Uso de CPU e memória $\leq 70\%$ em média;
  - Notificação de incidentes de hardware em menos de 5 minutos.

6. **Máquinas Virtuais (VMs) e Contêineres Docker**

- **Métricas:** *Uptime*, consumo de CPU, memória e disco, tempo de resposta das aplicações e ocorrência de reinícios inesperados.
- **SLIs:** *Uptime* das VMs e contêineres, latência de resposta dos serviços críticos e número de falhas.
- **SLOs propostos:**
  - Disponibilidade $\geq 99\%$ para VMs e $\geq 98\%$ para contêineres;
  - Resposta das aplicações < 60 ms;
  - No máximo 1 reinício inesperado por semana.

7. **Sites e Serviços Críticos**

- **Métricas:** Tempo de resposta HTTP/HTTPS, taxa de sucesso nas requisições, taxa de erros 4xx/5xx e disponibilidade externa.
- **SLIs:** Disponibilidade do serviço, latência média das requisições e taxa de erros em relação ao total de requisições.
- **SLOs propostos:**
  - Disponibilidade $\geq 99\%$;
  - Latência média < 100 ms;
  - Taxa de erros $\leq 1\%$.

Ao final desta etapa, foi estabelecido um conjunto estruturado de métricas, SLIs e SLOs que guiará o monitoramento do ambiente. Essa definição inicial possibilita identificar tanto dos pontos críticos quanto aqueles que estão desalinhados com a realidade.

4.1.3 Instrumentação da infraestrutura

4.1.3.1 Configuração das Ferramentas de Observabilidade

Com as métricas, SLIs e SLOs previamente definidos, a próxima etapa consistiu na configuração e integração das ferramentas escolhidas: Zabbix, Grafana e Graylog. O objetivo desta fase foi implementar um ambiente de monitoramento completo, capaz de identificar falhas rapidamente, emitir alertas automáticos e fornecer visibilidade em tempo real sobre a infraestrutura de TI.

A configuração foi dividida em três fases principais: implantação das ferramentas, integração e construção de *dashboards*.

4.1.3.2 Implantação e Configuração do Zabbix

O Zabbix foi escolhido como a ferramenta principal de monitoramento, responsável pela coleta de métricas de desempenho e disponibilidade em toda a infraestrutura. As principais ações realizadas foram:

- **Instalação e Configuração Inicial:**
  - Implantação do Zabbix Server 7.0 em uma máquina virtual dedicada;
  - Configuração do banco de dados para armazenamento histórico de métricas;
  - Criação de hosts para cada dispositivo identificado no levantamento do ambiente.
- **Coleta de Métricas:**
  - Utilização de agentes Zabbix quando possível;
  - Coletas SNMP (*Simple Network Management Protocol*)
  - *Checks* ICMP (*ping*) e HTTP/HTTPS para medir disponibilidade e latência de links e sites.
- **Definição de Triggers e Alertas:**
  - Criação de *triggers* para incidentes críticos com base nos SLOs definidos;
  - Configuração de notificações automáticas pelo Telegram;
  - Implementação de severidades distintas para alertas (informativo, aviso e crítico).

4.1.3.3 Implantação e Configuração do Graylog

O Graylog foi utilizado para centralização e análise de *logs*, permitindo correlação de eventos e detecção de comportamentos anômalos. A configuração incluiu:

- **Instalação e Configuração Inicial:**
  - Implantação do Graylog 6 em uma máquina virtual dedicada;
  - Configuração do banco de dados para armazenamento histórico de métricas;
- **Coleta de Logs:**
  - Recebimento de logs via Syslog de *firewalls*, switches e servidores;

4.1.3.4 Implantação e Configuração do Grafana

O Grafana foi responsável por centralizar a visualização dos dados coletados pelo Zabbix e Graylog, possibilitando análise em tempo real e acompanhamento do cumprimento dos SLOs.

- **Integração com Fontes de Dados:**
  - Conexão direta com o banco de dados do Zabbix;
- **Criação de Dashboards:**

- Painéis para monitoramento de *links* de internet, *firewall*, *switches* e *access points*;
- *Dashboards* de disponibilidade e consumo de recursos de servidores, VMs e contêineres;
- Visualização dos SLIs definidos, permitindo acompanhar o cumprimento dos SLOs em tempo real.
- Uso de gráficos históricos para identificar tendências e picos de utilização;

4.1.4 Coleta de Dados

Após a integração e automação do ambiente, iniciou-se a fase de coleta de dados. Esta etapa tem como objetivo validar os SLIs e SLOs definidos, medir a efetividade das ferramentas e avaliar o comportamento da infraestrutura sob condições de produção.

A coleta de dados foi realizada de forma contínua, contemplando o funcionamento normal do ambiente no dia a dia. Esse procedimento permitiu avaliar a capacidade do sistema de monitoramento em detectar, registrar e notificar incidentes de forma proativa.

Esta fase permitiu estabelecer métricas históricas e identificar possíveis *outliers*, fundamentais para diferenciar falhas reais de variações normais de operação.

4.1.5 Eliminação de TOIL

Nesta etapa, buscou-se reduzir o *TOIL*. A eliminação de *TOIL* tem como objetivo principal liberar a equipe de atividades rotineiras para que possa focar em tarefas de maior relevância estratégica, além de aumentar a confiabilidade e a eficiência operacional.

Por meio da análise dos fluxos operacionais e do histórico de incidentes analisados junto ao *Service Desk*, foram identificados os seguintes processos como passíveis de automação:

- **Reinício automático de serviços essenciais:** verificou-se que serviços críticos como a sincronização do AD (*Active Directory*) com alguns serviços externos e o serviço de coleta das marcações do relógio de ponto apresentavam interrupções ocasionais, afetando a disponibilidade de funcionalidades. Foi configurada uma automação no Zabbix para detectar a indisponibilidade e reiniciar automaticamente os serviços, reduzindo a intervenção manual e o tempo de indisponibilidade.
- **Coleta automática de contadores de impressoras:** anteriormente, a coleta de dados de consumo de impressões era realizada de forma manual, sujeita a erros e inconsistências. Desenvolveu-se um script em Python (Apêndice A) para executar a coleta de forma automática, armazenando os dados em um banco de dados centralizado. Essa abordagem possibilitou análises mais detalhadas, como a identificação de setores com maior consumo, variações mensais e a criação de indicadores para a gestão de custos.

- **Criação de API para redefinição de senha no AD:** para diminuir a dependência do time de TI em solicitações de suporte simples, está em estudo o desenvolvimento de uma aplicação interna, com interface web, que permite a redefinição de senha de usuários do AD mediante validação de informações pessoais, como CPF, data de nascimento e nome da mãe. Essa solução aumentará a autonomia dos usuários e reduzirá o volume de chamados relacionados a credenciais.

4.1.6 Dashboards dos dados obtidos

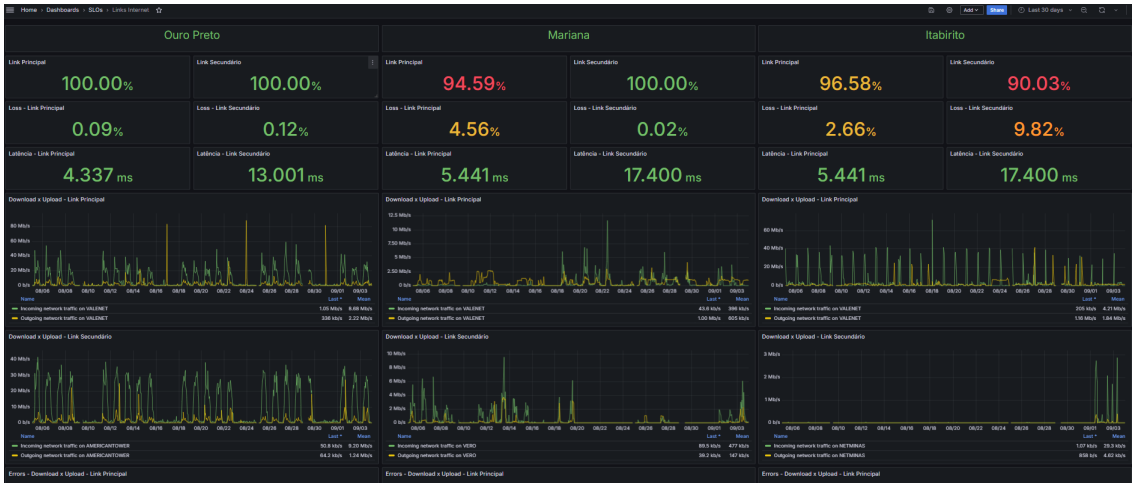


Figura 4.3 – *Dashboard* Links de Internet.
Fonte: Autoria própria

Figura 4.4 – *Dashboard* Links com as filiais
Fonte: Autoria própria

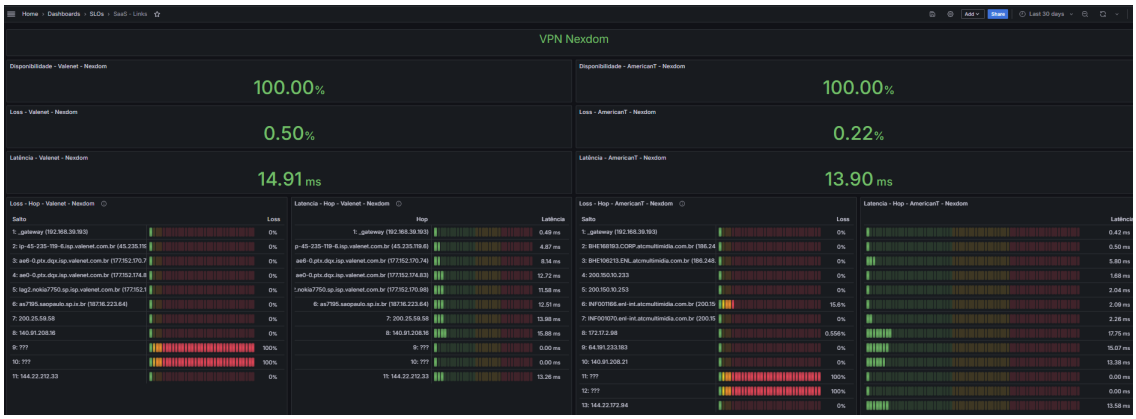


Figura 4.5 – *Dashboard* Links SaaS
Fonte: Autoria própria

Figura 4.6 – *Dashboard* Serviços SaaS
Fonte: Autoria própria

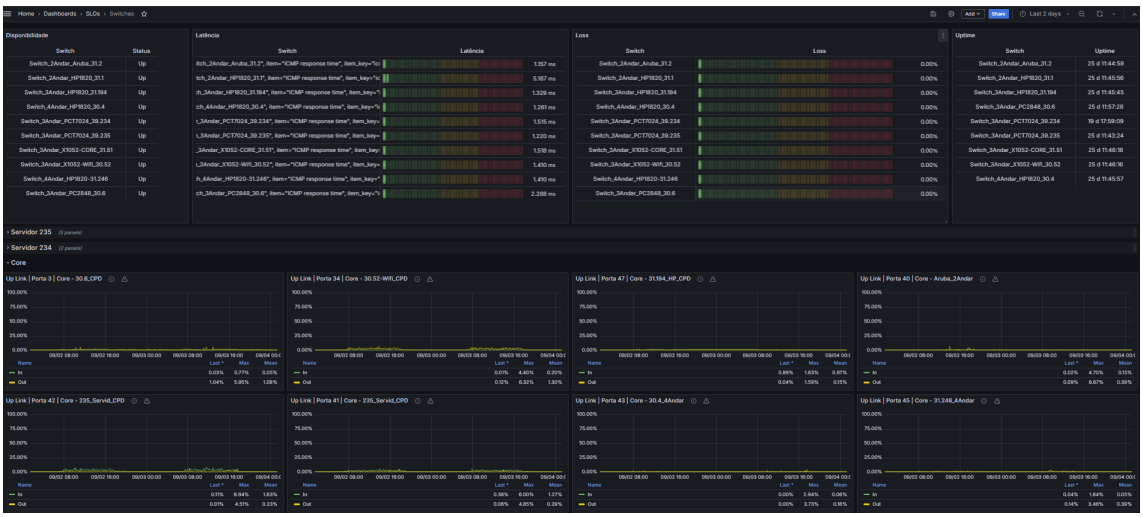


Figura 4.7 – *Dashboard* Switches
Fonte: Autoria própria

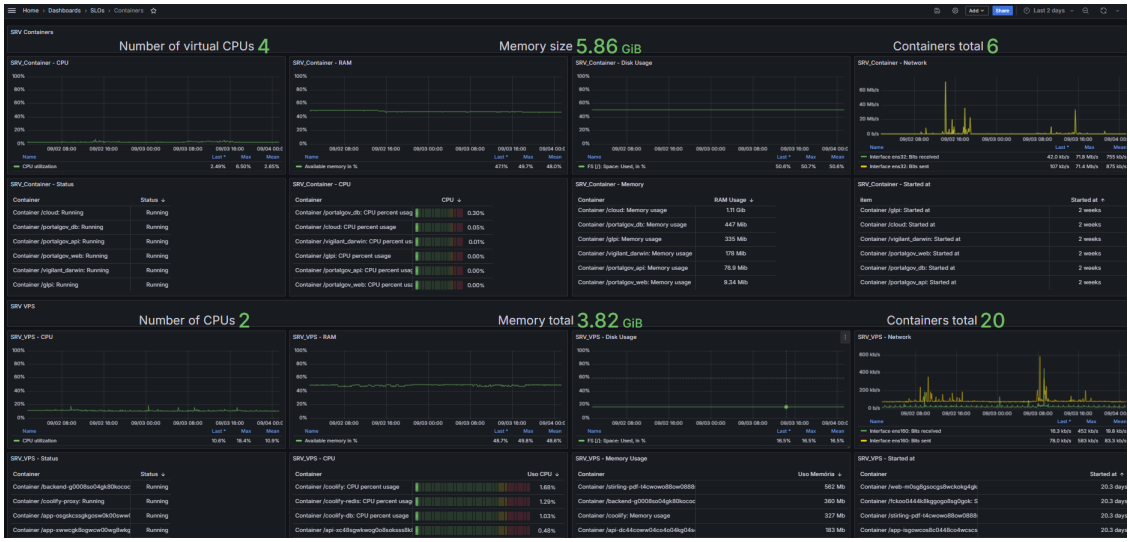


Figura 4.8 – *Dashboard* Containers
Fonte: Autoria própria

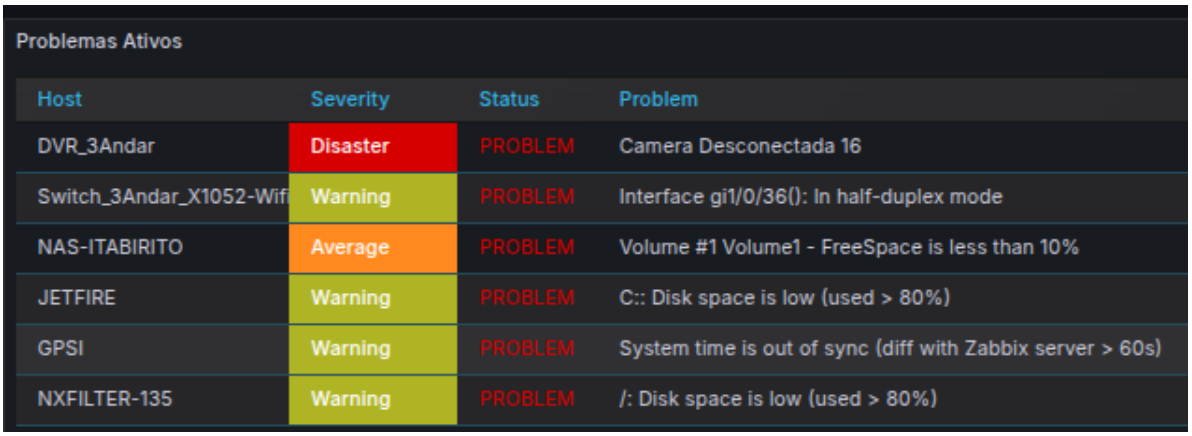


Figura 4.9 – *Dashboard* Problemas Ativos.
Fonte: Autoria própria

Figura 4.10 – *Dashboard* Falhas no login.
Fonte: Autoria própria

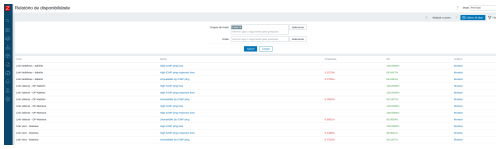


Figura 4.11 – Relatório de Disponibilidade do Zabbix.
Fonte: Autoria própria

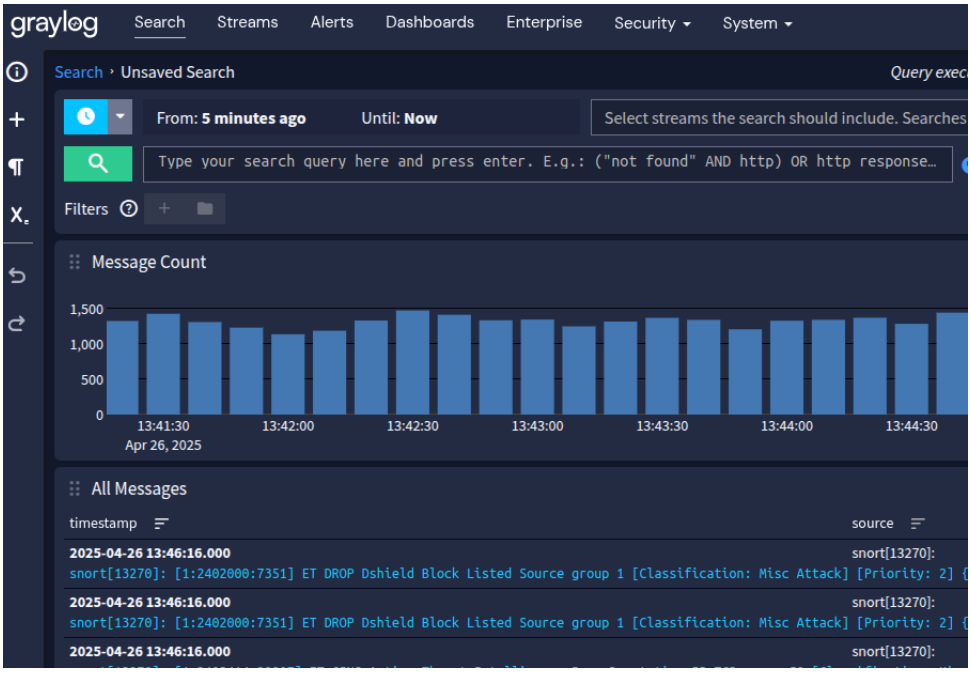
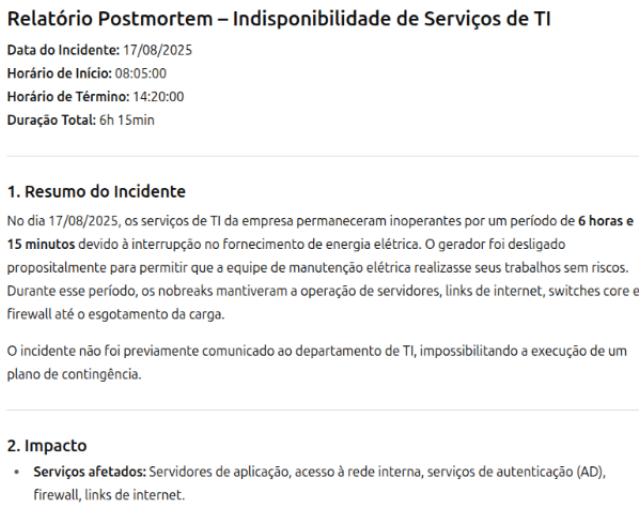


Figura 4.12 – Graylog.
Fonte: Autoria própria



Relatório Postmortem – Indisponibilidade de Serviços de TI

Data do Incidente: 17/08/2025
Horário de Início: 08:05:00
Horário de Término: 14:20:00
Duração Total: 6h 15min

1. Resumo do Incidente

No dia 17/08/2025, os serviços de TI da empresa permaneceram inoperantes por um período de **6 horas e 15 minutos** devido à interrupção no fornecimento de energia elétrica. O gerador foi desligado propositalmente para permitir que a equipe de manutenção elétrica realizasse seus trabalhos sem riscos. Durante esse período, os nobreaks mantiveram a operação de servidores, links de internet, switches core e firewall até o esgotamento da carga.

O incidente não foi previamente comunicado ao departamento de TI, impossibilitando a execução de um plano de contingência.

2. Impacto

- **Serviços afetados:** Servidores de aplicação, acesso à rede interna, serviços de autenticação (AD), firewall, links de internet.

Figura 4.13 – Postmortem.
Fonte: Autoria própria

5 Resultados

A análise dos dados obtidos ao longo do projeto permitiu avaliar de forma prática os impactos da implantação das práticas de SRE na infraestrutura de rede monitorada. Os resultados demonstram melhorias significativas tanto na capacidade de detecção de falhas quanto na resposta a incidentes críticos.

Em relação aos incidentes monitorados, os dados demonstraram que a detecção automática de falhas, associada à geração de alertas em tempo real, contribuiu de forma significativa para a redução dos chamados destinados à equipe de infraestrutura (Figura 5.1). Observou-se uma melhoria na capacidade de diagnóstico, uma vez que as informações coletadas permitiam a identificação rápida da natureza do problema. Como exemplo, alertas automáticos como notificações de toner baixo em impressoras permitiram ações proativas antes da interrupção dos serviços, contribuindo para praticamente zerar incidentes deste tipo.

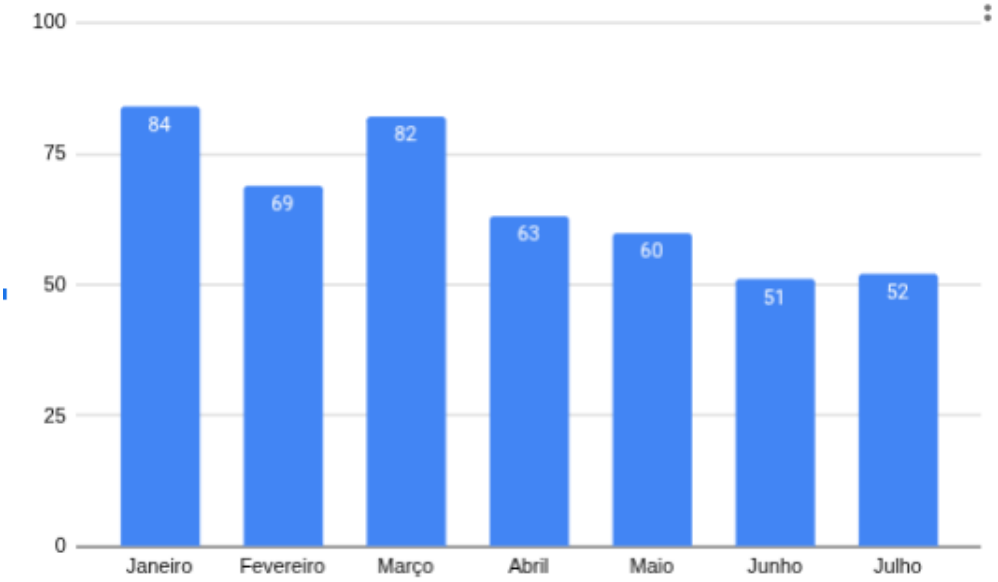


Figura 5.1 – Chamados abertos para o setor de infraestrutura por mês.
Fonte: *ServiceDesk*

Adicionalmente, a análise das métricas coletadas permitiu identificar inconsistências na alocação de recursos em máquinas virtuais. Algumas delas estavam subdimensionadas, comprometendo seu desempenho, enquanto outras encontravam-se superdimensionadas, consumindo recursos de forma desnecessária. Este diagnóstico possibilita uma redistribuição mais adequada da carga de trabalho e contribuiu para um melhor aproveitamento da infraestrutura existente. Como exemplo, a figura 5.3 apresenta o consumo de CPU, registrado nos últimos 90 dias para um dos servidores monitorados. Observa-se que, nesse período, o pico de utilização atingiu apenas 36%, com uma média de consumo em torno de 3,0%, evidenciando que a máquina está

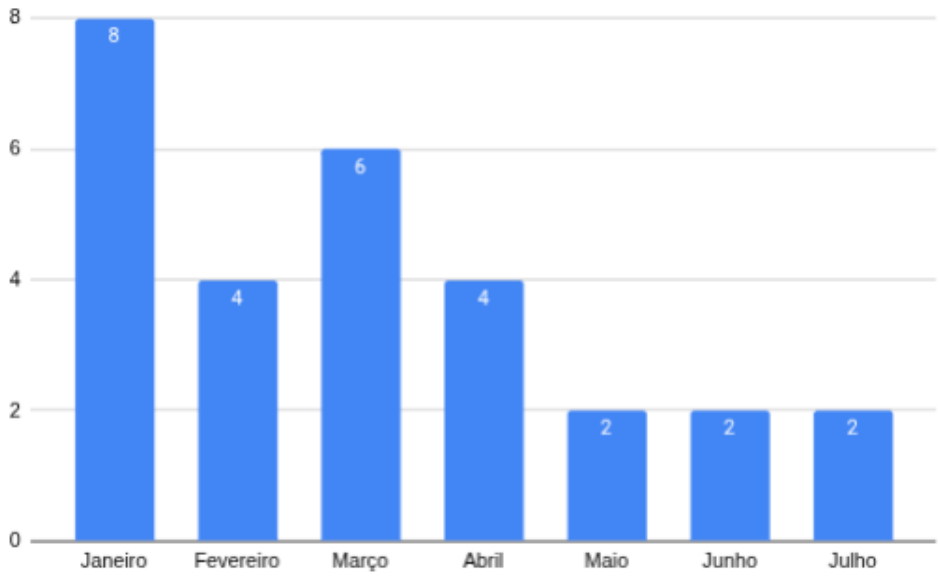


Figura 5.2 – Chamados abertos problemas impressora.
Fonte: *ServiceDesk*

superdimensionada em relação à sua carga real de trabalho.

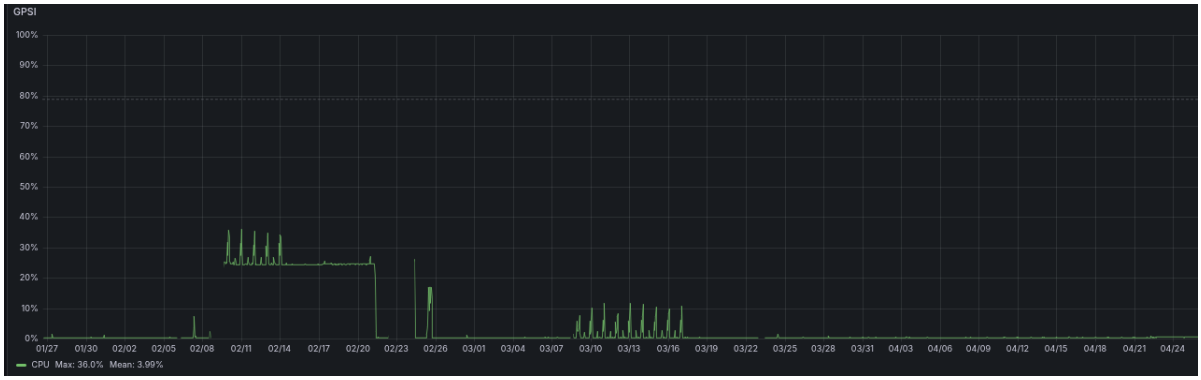


Figura 5.3 – *Dashboard* Monitoramento do uso de CPU.
Fonte: Autoria própria

Além disso, foi possível estimar a taxa de crescimento do volume de dados destinados a backup, o que forneceu subsídios importantes para um planejamento de capacidade mais preciso e preventivo. Essa necessidade tornou-se ainda mais crítica após a determinação institucional de que as imagens médicas armazenadas devem ser preservadas por um período mínimo de 20 anos. Por meio do monitoramento contínuo, observou-se que, ao longo de um ano, houve um incremento de aproximadamente 300 GB no consumo do espaço em disco destinado ao armazenamento desses dados. Com essas informações, foi possível calcular a taxa média de crescimento anual e projetar a necessidade futura de armazenamento, garantindo que as políticas de retenção estabelecidas sejam cumpridas sem comprometer a operação do sistema.

Quanto ao uso do Graylog, a centralização dos logs possibilitou identificar comportamentos anômalos, como tentativas recorrentes de autenticação falha entre usuários específicos do

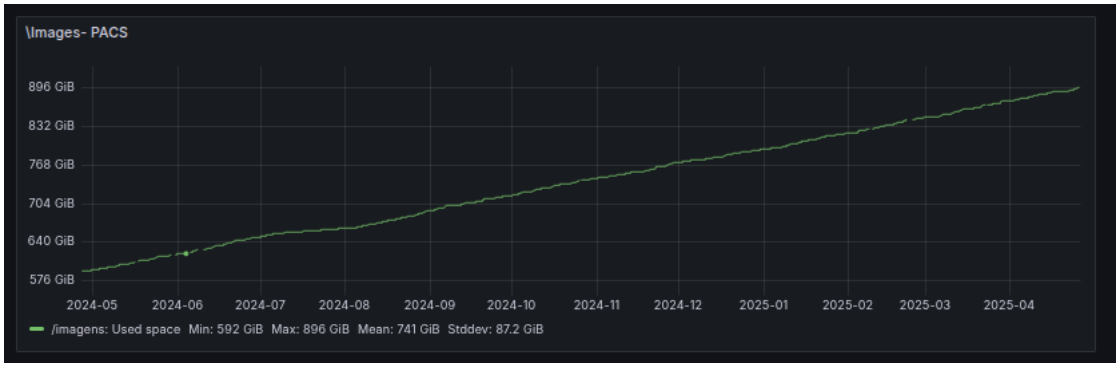


Figura 5.4 – *Dashboard* Armazenamento da partição de imagens.
Fonte: Autoria própria

Figura 5.5 – *Dashboard* Falhas login por usuário .
Fonte: Autoria própria

domínio. Chegando a conclusão que existia na infraestrutura *softwares* tentando logar automaticamente em alguns serviços.

A coleta estruturada e consolidada desses dados facilitou análises mais rápidas e eficazes, fortalecendo a postura de segurança da organização e contribuindo para a identificação precoce de falhas operacionais.

De maneira geral, a análise dos dados confirmou a eficácia da instrumentação implementada e evidenciou o impacto positivo da observabilidade na resiliência e estabilidade da infraestrutura de rede. Ficou clara a importância de um monitoramento contínuo e estruturado para antecipação de falhas e mitigação de riscos operacionais.

Ainda, está em andamento o estudo de alternativas para a integração do Graylog ao Grafana, com o objetivo de consolidar ainda mais os dados de monitoramento em uma plataforma única, o que poderá trazer ganhos adicionais em agilidade, eficiência e capacidade analítica no futuro.

Por fim, a consolidação dos resultados permitiu identificar pontos de melhoria, como a necessidade de ajustes nos gatilhos de monitoramento para reduzir a ocorrência de alertas falsos positivos e a inundação de notificações, buscando tornar o ambiente de monitoramento ainda mais preciso, inteligente e alinhado às necessidades operacionais da organização.

6 Conclusão

Partindo de um cenário em que quase não existia qualquer tipo de monitoramento, a adoção de práticas de SRE na infraestrutura de rede da organização revelou-se altamente satisfatória para a detecção, resposta e prevenção de incidentes. A implementação de ferramentas como Zabbix, Graylog e Grafana possibilitou a transformação de um ambiente até então reativo em um ambiente proativo e resiliente, com um nível significativamente mais elevado de maturidade em gestão de rede.

Os resultados demonstraram que a implementação das ferramentas de observabilidade trouxe benefícios tangíveis para a operação da organização, como a redução do tempo de resposta a falhas, a melhoria na capacidade de diagnóstico e a antecipação de problemas que poderiam impactar a disponibilidade dos serviços. Além disso, a consolidação das métricas coletadas possibilitou ações de otimização de recursos, planejamento de capacidade e fortalecimento da postura de segurança da infraestrutura de TI.

Como trabalhos futuros, destaca-se a possibilidade de ampliar a aplicação da observabilidade para além do monitoramento técnico, incorporando também indicadores de negócio. Indicadores relacionados ao negócio, como por exemplo, o tempo de espera do cliente para receber atendimento, pode gerar informações do serviço prestado. A análise desses dados possibilita a identificação de gargalos, e consequentemente a melhoria destes serviços. Contribuindo para aprimorar a experiência do cliente e, em última instância, fortalecer a imagem institucional da empresa.

Referências

AGUIAR, I. F. *Proposta De Utilização Da Ferramenta Zabbix No Gerenciamento De Redes: Um Estudo De Caso No Ambiente Da FAB Segundo Boas Práticas De Governança De Ti*. [S.l.], 2013. Acessado em 22 de abril de 2025. Disponível em: <https://pantheon.ufrj.br/bitstream/11422/3300/4/IAguiar.pdf>.

AZAMBUJA, R. G. P. *Gerenciamento Da Rede: Infraestrutura E Monitoramento Do Hospital Universitário De Santa Maria (Husm)Network Management: Infrastructure And Monitoring Of Hospital Universitário De Santa Maria (Husm)*. [S.l.], 2023. Acessado em 22 de abril de 2025. Disponível em: <https://periodicos.ufn.edu.br/index.php/disciplinarumNT/article/view/4534/3193>.

BEYER C. JONES, J. P. R. M. B. *Site Reliability Engineering*. [S.l.]: O'Reilly, 2016.

CLEMM, A. *Network Management Fundamentals*. [S.l.]: Cisco Press, 2006.

CLIMENT, J. *How maintenance windows affect your error budget—SRE tips*. [S.l.], 2020. Acessado em 22 de julho de 2025. Disponível em: <https://cloud.google.com/blog/products/management-tools/sre-error-budgets-and-maintenance-windows>.

DAVE, D. M. Impact of site reliability engineering on manufacturing operations improving efficiency and reducing downtime. *International Journal of Scientific and Research Publications*, v. 13, 2023. Acessado em 13 de julho de 2025. Disponível em: <https://www.researchgate.net/profile/Deep-Manishkumar-Dave/publication/375484574_Impact_of_Site_Reliability_Engineering_on_Manufacturing_Operations_Improving_Efficiency_and_Reducing_Downtime/links/654b871c3fa26f66f4e738b6/Impact-of-Site-Reliability-Engineering-on-Manufacturing-Operations-Improving-Efficiency-and-Reducing-pdf>.

FUZI, M. F. M. *Integrated Network Monitoring Using Zabbix With Push Notification Via Telegram*. [S.l.], 2022. Acessado em 22 de abril de 2025. Disponível em: <https://jcrinn.com/index.php/jcrinn/article/view/282/195>.

Grafana Labs. *Grafana Documentation*. [S.l.], 2025. Acessado em: 8 abr. 2025. Disponível em: <https://grafana.com/docs/grafana/latest/>.

Graylog, Inc. *Graylog Documentation*. [S.l.], 2025. Acessado em: 8 abr. 2025. Disponível em: <https://go2docs.graylog.org/>.

IEEE. 1990. Disponível em: <http://www.informatik.htw-dresden.de/~hauptman/SEI/IEEE_Standard_Glossary_of_Software_Engineering_Terminology%20.pdf>. Acessado em: 12 de agosto de 2024.

KáLMáN, R. E. *On the general theory of control systems*. [S.l.: s.n.], 1960.

Lessa, Demian. *O Protocolo de Gerenciamento RMON*. [S.l.], 1999. Acessado em 19 de março de 2025. Disponível em: <https://memoria.rnp.br/newsgen/9901/rmon.html>.

MCGILLICUDDY, S. 2022. Disponível em: <https://www.ibm.com/downloads/cas/MPJ0GVXZ>. Acessado em 30 de setembro de 2024.

MUNIZ A. ARANHA, F. S. T. P. W. L. A. *Jornada da Observabilidade*. [S.l.]: Brasport, 2024.

PEIXIAN, C. *Research on Cluster Monitoring and Prediction Platform based on Zabbix Technology*. [S.l.], 2020. Acessado em 24 de abril de 2025. Disponível em: <https://iopscience-iop-org.ez28.periodicos.capes.gov.br/article/10.1088/1755-1315/512/1/012155/pdf>.

PHALTANKAR, K. *Practical Guide for Implementing Secure Intranets and Extranets. 1.ed.* [S.l.]: Artech House Publishers, 1999.

SILVA, E. P. *Proposta De Plano De Continuidade De Negócio Em Ti Para Um Hospital Norte Catarinense*. [S.l.], 2021. Acessado em 22 de abril de 2025. Disponível em: <https://repositorio-api.animaeducacao.com.br/server/api/core/bitstreams/64c1edb2-2e79-410f-85b4-deab38c09c99/content>.

SILVA, W. J. d. S. R. A. da. *A Implementação Do Zabbix Com Segurança: Um Estudo De Caso Zabbix Safely*. [S.l.], 2024. Acessado em 22 de abril de 2025. Disponível em: <https://ojs.focopublicacoes.com.br/foco/article/view/4851/3459>.

SOUZA, M. C. de. *Engenharia de Confiabilidade de Sites: Aplicabilidade do SRE nas empresas de tecnologia*. [S.l.], 2024. Acessado em 19 de julho de 2025. Disponível em: <https://ric.cps.sp.gov.br/bitstream/123456789/33456/1/analiseedesenvolvimentodesistemas_2024_1_matheuscardosodesouza_engenhariadeconfiabilidadedesites.pdf>.

STALLINGS, W. *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2*. [S.l.]: Addison-Wesley, 1998.

STALLINGS, W. *Cryptography and Network Security: Principles and Practice*. [S.l.]: Pearson, 2016.

USMAN, M. 2022. Disponível em: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9837035>. Acessado em 23 de setembro de 2024.

Zabbix LLC. *Zabbix Documentation*. [S.l.], 2025. Acessado em: 8 abr. 2025. Disponível em: <https://www.zabbix.com/documentation/current/en/manual>.

APÊNDICE A – Metodologia

Listing A.1 – Script desenvolvido em Python no Jupyter Notebook

```
# Instalando bibliotecas necessarias

! pip install pyzabbix
! pip install pandas openpyxl ipykernel
! pip install python-dotenv

# Variaveis utilizadas para conexao ao Zabbix
import os
from dotenv import load_dotenv

load_dotenv()

ZBX_URL = os.getenv("ZBX_URL")
ZBX_USERNAME = os.getenv("ZBX_USERNAME")
ZBX_PASSWORD = os.getenv("ZBX_PASSWORD")

# Conectando na API Zabbix
from pyzabbix import ZabbixAPI

def connect_zabbix():
    #load_dotenv()
    api_url = ZBX_URL
    api_username = ZBX_USERNAME
    api_password = ZBX_PASSWORD

    try:
        zapi = ZabbixAPI(api_url)
        zapi.login(api_username, api_password)
        print(f'Conectado ao Zabbix: {zapi.api_version()}')
        return zapi

    except Exception as e:
        print(f'Erro ao conectar ao Zabbix: {e}')
```

```
        return exit(1)

# Lista Grupos de Hosts
def listar_grupos_zabbix():
    zapi = connect_zabbix()
    try:
        grupos = zapi.hostgroup.get(
            output = ["groupid", "name"]
        )
        return grupos

    except Exception as e:
            print(f"Erro ao grupos de hosts: {e}")
            return None

grupos = listar_grupos_zabbix()
for grupo in grupos:
    print(grupo)

# Objeto Host
class Host:
    def __init__(self, host_id, host_name, host_description, host_type,
        self.host_id = host_id
        self.host_name = host_name
        self.host_description = host_description
        self.host_type = host_type
        self.host_department = host_department

    def __str__(self):
        return f"Host ID: {self.host_id}, Nome: {self.host_name}, Descr

    def get_host_name(self):
        return self.host_name

# Grupos das impressoras
groupid = 29

# Conexao com Zabbix
zapi = connect_zabbix()
```

```
# Funcao auxiliar para extrair e separar os valores da opcao TAG (Etiqu
def get_tag_value(tags, tag_name):
        """Extrai o valor de uma tag especifica."""
        for tag in tags:
            if tag['tag'] == tag_name:
                return tag['value']
        return None


# Get Hosts — Somente Hosts Ativos
# Retorna hosts
def get_hosts(zapi, groupid):
    try:
        all_hosts = zapi.host.get(
            groupids=groupid,
            output=["hostid", "host", "name", "status", "description"],
            selectTags="extend"
            )
        # Pega somente os hosts que estao com Status —> Ativo
        hosts = [h for h in all_hosts if int(h['status']) == 0]

        return hosts

    except Exception as e:
        print(f"Erro ao listar hosts: {e}")
        return []

# Cria lista com hosts (Classe Host)
# Retorna lista de objetos hosts
hosts = get_hosts(zapi, groupid)
def cria_lista_hosts(hosts):
    hosts_list = []
    for host in hosts:
        host_obj = Host(
            host_id = host['hostid'],
            host_name = host['name'],
            host_description = host['description'],
```

```
            host_type = get_tag_value(host['tags'], 'Tipo'),
            host_department = get_tag_value(host['tags'], 'Setor')
        )
        hosts_list.append(host_obj)

    return hosts_list

# Print da lista dos hosts lidos
list_hosts = cria_lista_hosts(hosts)
for host in list_hosts:
     print(host)

# Objeto Item

class Item:
    def __init__(self, item_id, item_hostid, item_key, item_name, item_
        self.item_id = item_id
        self.item_hostid = item_hostid
        self.item_key = item_key
        self.item_name = item_name
        self.item_lastclock = item_lastclock
        self.item_lastvalue = item_lastvalue

    def __str__(self):
        return f"Item ID: {self.item_id}, Host ID: {self.item_hostid},

# Conexao com Zabbix
zapi = connect_zabbix()
# Conecta ao Zabbix e Le os itens
# Chama cria lista_itens para criar lista de objetos Item
def get_itens(host_id):
    try:
        itens = zapi.item.get(
            hostids=[host_id],
            output=["itemid", "key_", "name", "lastclock", "lastvalue",
        )
        return cria_lista_itens(host_id, itens)

    except Exception as e:
```

```
        print(f"Erro ao ler itens: {e}")
        return []


# A funcao acima chama essa funcao para criar objetos Item
# Retorna lista de objetos Item
def cria_lista_itens(hostid, itens):
    item_list = []
    for item in itens:
        if item['status'] == '0':
            item_obj = Item(
                item_id = item['itemid'],
                item_hostid = hostid,
                item_key = item['key_'],
                item_name = item['name'],
                item_lastclock = item['lastclock'],
                item_lastvalue = item['lastvalue']
            )
            item_list.append(item_obj)
    return item_list

# A partir da lista de objetos Item separa quais itens sao Serial Numbe
# Compara os nomes dos itens com palavras-chave para identificar cada t
# Retorna serial, contador preto e contador colorido
def processa_itens(itens):
    serial = ''
    contador_pb = 0
    contador_c = 0

    for item in itens:
        if 'serial' in item.item_name.lower():
            serial = item.item_lastvalue
        elif any(pb in item.item_key.lower() for pb in ['pb', 'pbkm', '
            contador_pb = item.item_lastvalue
        elif any(color in item.item_key.lower() for color in ['contador
            if 'ckmax' in item.item_key.lower():
                contador_c = None
            else:
                contador_c = item.item_lastvalue
```

```
    print(f'Itens processados: {serial} | {contador_pb} | {contador_c}'

    return serial, contador_pb, contador_c

def mes_anterior():
    """Retorna o mes anterior no formato MM/AAAA."""
    from datetime import datetime, timedelta
    mes_atual = datetime.now()
    mes_anterior = mes_atual - timedelta(days=30)  # Aproximacao de 30
    return mes_anterior.strftime("%m/%Y")

# Define classe Impressora
from datetime import datetime
class Printer:
    def __init__(self, host_id, host_name, serial_number, contador_pb,
        self.host_id = host_id
        self.host_name = host_name
        self.serial_number = serial_number
        self.contador_pb = contador_pb
        self.contador_c = contador_c
        self.competencia = competencia
        self.data_update = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        self.status = 1
    def __str__(self):
        return f"Host ID: {self.host_id}, Nome: {self.host_name}, Seria

# Processa os itens de cada host e cria objetos Printer
# Retorna lista de objetos Printer

def list_printers():
    list_printers = []

    for host in list_hosts:
        name = host.get_host_name()
        itens = get_itens(host.host_id)
        list_aux = []
        for item in itens:
            list_aux.append(item)
```

```
            serial, contador_pb, contador_c = processa_itens(list_aux)
            printer = Printer(
                host_id=host.host_id,
                host_name=name,
                serial_number=serial,
                contador_pb=contador_pb,
                contador_c=contador_c,
                competencia=mes_anterior()

            )

            print(printer)

            list_printers.append(printer)
        return list_printers

# Funcao que formata o valor da coluna [Serial].
# Divide impressoras coloridas em 2 itens
#
# Recebe uma tupla contendo (serial, contador_pb, contador_c)
#   Se Contador C for None ou 0, adiciona somente o contador PB
#   Se Contador C for diferente de None ou 0, adiciona dois itens:
#       - Contador C com sufixo '-C'
#       - Contador PB com sufixo '-PB'

def formata_serial(list_printers_list):
    formatted_list = []
    for printer in list_printers_list:
        cont_c = printer.contador_c
        #print(printer)
        #print('-'*20)
        if printer.contador_c is None or printer.contador_c == 0:
            #print(f"Contado none ou 0 --> {printer.contador_c}")
            #printer.contador_c = 0
            formatted_list.append({
                'Host ID': printer.host_id,
                'Nome': printer.host_name,
                'Serial_Number': printer.serial_number,
                'Contador': printer.contador_pb,
```

```
                #'Contador C': printer.contador_c,
                'Competencia': printer.competencia,
                'Data Update': printer.data_update
            })
        else:
            #print(f'Contador diferente de none ou 0 --> {printer.conta
            # Adicona na lista um item para o contador colorido
            # Serial passa a conter o sufixo '-C
            formatted_list.append({
                'Host ID': printer.host_id,
                'Nome': printer.host_name,
                'Serial_Number': printer.serial_number + '-C',
                #'Contador PB': printer.contador_pb,
                'Contador': printer.contador_c,
                'Competencia': printer.competencia,
                'Data Update': printer.data_update
            })

            formatted_list.append({
                'Host ID': printer.host_id,
                'Nome': printer.host_name,
                'Serial_Number': printer.serial_number + '-PB',
                'Contador': printer.contador_pb,
                #'Contador C': printer.contador_c,
                'Competencia': printer.competencia,
                'Data Update': printer.data_update
            })

    return formatted_list

import pandas as pd
def salvar_printers_em_excel(printers, nome_arquivo):
    # Converter a lista de objetos Printer em lista de dicionarios
    print('Lista Printers')
    for i in printers:
        print(i)
        print(i.get('Serial_Number'))
    print('-'*80)
    dados = [
```

```
        {
            "Serial": p.get("Serial_Number"),
            "Nome": p.get("Nome"),
            "Contador": p.get("Contador"),
            #"Contador_C": p.contador_c
            "Competencia": p.get("Competencia"),
            "Data Update": p.get("Data_Update", datetime.now().strftime
        }
        for p in printers
    ]

    # Criar DataFrame e salvar em Excel
    df = pd.DataFrame(dados)
    df.to_excel(nome_arquivo, index=False)
    print(f"Arquivo salvo como: {nome_arquivo}")

# Transforma lista de impressoras em lista de objetos Printer
printers_lista = list_printers()

# Formata a coluna [Serial] e divide em dois itens se necessario
printers_f = formata_serial(printers_lista)


salvar_printers_em_excel(printers_f, nome_arquivo="printers.xlsx")

# Processa planilha mes anterior

def calcular_valor(df):
    def calcular_por_serial(row):
        if isinstance(row["Serial"], str) and row["Serial"].endswith("-C
            return row["Total de copias"] * 0.8
        else:
            return row["Total de copias"] * 0.07

    df["Valor"] = df.apply(calcular_por_serial, axis=1)
    return df

def adiciona_total(df):
    total_geral = df["Total de copias"].sum()
```

```
    print(f"Total Geral: {total_geral}")
    nova_linha = {col: "" for col in df.columns}
    nova_linha["Total"] = total_geral
    df.loc[len(df)] = nova_linha
    df.iloc[-1, 0] = "Total:"  # coloca "Total:" na primeira coluna
    return df


def formata_colunas(df):
    # Supondo que as colunas ja existam:
    colunas = df.columns.tolist()

    # Remover "Contador Final" da lista e inserir logo apos "Contador I
    colunas.remove("Contador Final")
    indice = colunas.index("Contador Inicial") + 1
    colunas.insert(indice, "Contador Final")

    # Reorganizar o DataFrame
    df = df[colunas]

    # Deletar a ultima coluna
    df = df.iloc[:, :-1]

    return df

# Recebe um DataFrame com os contadores e atualiza a coluna "Contador F

def atualizar_contadores(df_contadores):
    df_atual = pd.read_excel('printers.xlsx')

    # Garantir que a coluna Serial seja string e remova espacos em bran
    df_contadores["Serial"] = df_contadores["Serial"].astype(str)
    df_contadores["Serial"] = df_contadores["Serial"].str.strip()
    df_atual["Serial"] = df_atual["Serial"].astype(str)
    df_atual["Serial"] = df_atual["Serial"].str.strip()

    # Reduz o df_atual para apenas os campos necessarios
    df_atual_reduzido = df_atual[["Serial", "Contador"]]
```

```
    # Junta os dois dataframes com base em Serial (left join para mante
    df_resultado = df_contadores.merge(df_atual_reduzido, on="Serial",

    # Preenche valores ausentes com 0 na nova coluna "Contador"
    df_resultado["Contador"] = df_resultado["Contador"].fillna(0).astyp
    # Renomeia a coluna "Contador" para "Contador Final"
    df_resultado = df_resultado.rename(columns={"Contador": "Contador F

    df_resultado.to_excel("contadores_atualizados.xlsx", index=False)

    return df_resultado


# Recebe xlsx de entrada com os campos:
#     ["Serial", "Modelo", "Local", "IP", "Contador Inicial", "Contador
#     - Deleta coluna "Contador Inicial" e renomeia "Contador Final" pa
#     - Atualiza os valores de "Contador Final" com os valores retornad
def processar_planilha(arquivo_entrada, arquivo_saida):
    # Le a planilha e seleciona a aba "25-Abril"
    # Descobrir todas as abas
    print('-'*80)

    abas = pd.ExcelFile(arquivo_entrada).sheet_names
    print(f'Abas encontradas: {abas}')
    # Selecionar a ultima aba
    ultima_aba = abas[-1]
    df = pd.read_excel(arquivo_entrada, sheet_name=ultima_aba,skiprows=
    #print(df)

    df = df.drop(df.columns[[11, 12, 13]], axis=1)
    df = df.drop(columns=["Contador Inicial"], axis=1)
    df = df.rename(columns={"Contador Final": "Contador Inicial"})
    df = df.iloc[:, :-1]
    #display(df)
    # Atualiza os contadores com os valores mais recentes
    df = atualizar_contadores(df)
    #display(df)
```

```
    df["Total de copias"] = df["Contador Final"] - df["Contador Inicial
    df = formata_colunas(df)
    df = calcular_valor(df)
    display(df)
    # Selecionar apenas as colunas desejadas para o novo arquivo
    df_saida = df[["Serial", "Modelo", "Local", "IP", "Contador Inicial
    #df_saida = adiciona_total(df_saida)
    # Salvar em novo arquivo .xlsx
    df_saida.to_excel(arquivo_saida, index=False)
    total_copias = df["Total de copias"].sum()
    valor_copias = df["Valor"].sum()
    print(f"Total de copias: {total_copias}")
    print(f"Valor total: {valor_copias}")

# Extrai local e sala do nome do host
def extrair_local_sala(texto):
    partes = texto.split('_')
    if len(partes) >= 3 and partes[0] == "Print":
        local = partes[1]
        sala = partes[2]
        return local, sala
    else:
        raise ValueError("Formato invalido. Esperado: 'Print_Local_Sala

processar_planilha("2025-07.xlsx", "_Saida_Contadores.xlsx")
```